PL-TR-95-2083

PROGRAMMER AND END USER GUIDE FOR PROCESSING LEPA DATA FILES

J. M. Griffin
J. N. Bass

Radex, Inc.
Three Preston Court
Bedford, MA 01730

May 3, 1995

Scientific Report #5

Approved for public release; distribution unlimited

PHILLIPS LABORATORY
Directorate of Geophysics
AIR FORCE MATERIEL COMMAND
HANSCOM AIR FORCE BASE, MA 01731-3010

"This technical report has been reviewed and is approved for publication"

EDWARD C. ROBINSON
Contract Manager
Data Analysis Division

ROBERT E. McINERNEY, Director
Data Analysis Division

This report has been reviewed by the ESD Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS).

Qualified requestors may obtain additional copies from the Defense Technical Information Center. All others should apply to the National Technical Information Service.

If your address has changed, or if you wish to be removed from the mailing list, or if the addressee is no longer employed by your organization, please notify PL/IM, 29 Randolph Road, Hanscom AFB, MA 01731-3010. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document requires that it be returned.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	3 May 1995	Scientific Report #5

4. TITLE AND SUBTITLE
Programmer and End User Guide
for Processing LEPA Data Files

5. FUNDING NUMBERS
PE 62101F
PR 7601 TA 22 WU RA

Contract F19628-90-C-0090

6. AUTHOR(S)
J. M. Griffin
J. N. Bass

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
RADEX, Inc.
Three Preston Court
Bedford, MA 01730

8. PERFORMING ORGANIZATION REPORT NUMBER
RXR-95061

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Phillips Laboratory
29 Randolph Road
Hanscom AFB, MA 01731-3010

Contract Manager: Edward C. Robinson/GPD

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
PL-TR-95-2083

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for Public Release
Distribution Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Several PC-based FORTRAN codes are available for processing data files generated by both the scientific magnetometer and Low Energy Plasma Analyzer (LEPA), instruments mounted on the Combined Release and Radiation Effects Satellite (CRRES). These codes read in the raw and background counts and convert the net counts into differential directional number fluxes, $j(\Omega,E)$, for each specific sensor look direction as functions of voltage-level mode and energy channel. These codes also use the magnetometer data to derive pitch angles, α, for the corresponding fluxes. LEPAFLUX uses a PL post-processed magnetometer file to find $j(\Omega,E)$ and α in satellite body-centered coordinate systems. PCCALMAG derives the ECI-based components of the measured geomagnetic field with higher time resolution to be used by GET3DBIN. Like LEPAFLUX, the GET3DBIN code generates $j(\Omega,E)$. GET3DBIN then uses the attitude determination software previously developed by Radex to transform the look directions of the measurements to a space-fixed system in which the fluxes are sorted by α and azimuth, defined as the angle of rotation about the magnetic field line from a direction approximately radially outward. GET3DBIN lastly writes out the mean fluxes and the associated bin indices to a file. This guide will describe the software processing for extracting the three-dimensional distribution function of the electrons and ions with energy range from 10 eV to 30 keV. Beside giving the operational procedures of the codes, the guide will also provide physical insights and mathematical background of the algorithms whenever appropriate for better understanding of the outputs.

14. SUBJECT TERMS
LEPA, CRRES, Differential directional number flux, Pitch angle, 3-Dimensional plasma distribution, Magnetometer, VDH, ECI to VDH coordinate transformation

15. NUMBER OF PAGES
60

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Unlimited

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont'd)

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

The work described in this report required the involvement and guidance of a number of individuals at Phillips Laboratory, and their interest and encouragement is gratefully acknowledged.

The authors wish to thank Dave Hardy of PL/GPSG for initiation of this work, and his valuable guidance in the course of the projects. The authors also wish to thank Rick Lambour of PL/GPSG, for his leadership and interest in this work. The acknowledgement is also extended to Ernie Holeman, Captain Mike Violet, and First Lieutenant Jabin Bell, all of PL/GPSP, for assistance in collection of the appropriate data files and supporting software codes.

The authors also wish to thank Bill McNeil of Radex for his advice and assistance with the codes relating to the CRRES attitude determination and the processing of the CRRES science magnetometer data. The authors also acknowledge Howard Singer, formerly of PL/GPSG, who was the Principal Investigator of the CRRES science magnetometer. The authors thank Ed Robinson, PL/GPD, who, as Contract Monitor, coordinated the work and provided valuable guidance. Doug Reynolds of Radex is also hereby acknowledged for his contributions to the programming efforts and comments on the same.

1. INTRODUCTION

There are data files containing measurements made by the scientific magnetometer and the Low Energy Plasma Analyzer (LEPA), two of the instrument packages flown on the Combined Release and Radiation Effects Satellite (CRRES) mission. Several PC-based software codes have been written to numerically derive the three-dimensional profile of electron and ion plasma density from piecewise post-processing of the aforementioned data files. The purpose of this report is to provide assistance in running those codes and to furnish essential information for better understanding of the outputs from the programs. The initial discussion in this section will focus on the general operation of the LEPA instrument. Immediately following is an overview of the theoretical concepts incorporated into the software packages. Next, the basic steps in the software processing will be outlined. The final paragraph in this section will be a brief review on the subsequent sections where detailed descriptions of the programs are provided.

Other documentation is available that provides a more detailed description of LEPA. The basic principles of the LEPA design and operation are described in *Hardy, et al.*, [1993a]. Considerable details on electronic operations and calibrations of the LEPA are available in *Hardy, et al.* [1993b]. Outlined discussion on algorithms needed to process the LEPA datafiles is given in a supporting document [*Kerns*, 1993]. With the wealth of information available, this guide will only describe how to use the software codes developed by Radex for processing the particle data collected by LEPA and the geomagnetic field measurement made by the scientific magnetometer. There will be some discussion on the theoretical motivations and the basic background details necessary to fully understand the outputs of the software codes. The enclosed text will provide step by step details of the software processing from determining the measured geomagnetic field components to binning the differential directional fluxes for extracting the three-dimensional distribution functions.

1.1 BACKGROUND INFORMATION ON LEPA

CRRES was launched in July 1990 with the planned 5-year mission to sample radiation levels by detecting charged particles, and to measure geomagnetic fields over a large portion of near-Earth space. The satellite occasionally ejected canisters filled with materials and measured the resulting induced radiation effects. At other times, onboard instrumentation observed a nominal radiation environment. Two types of experiments were conducted onboard the satellite. The majority of the experiments were designated as being of the Geosynchronous Transfer Orbit (GTO) class; the remainder relegated to the Low Altitude Satellite Studies of Ionospheric Irregularities (LASSII) class. CRRES orbited Earth in a low-inclination geosynchronous transfer orbit collecting and transmitting data until a battery malfunction cut off the telemetry link permanently in October of 1991. Among the experiments and instrument packages on the satellite was LEPA, the key instrument relevant to this guide.

LEPA is capable of detecting charged particles with energy ranging from 10 ev to 30 kev. The

device is actually a pair of tri-quadraspherical electrostatic analyzers; each analyzer is designed to sample only the incident particles within a spatial resolution defined by an operational mode and within a certain energy range selected by a voltage mode. The energy of the particle being detected at a given time is a function of the voltage difference between concentric spherical sections. This voltage difference is controlled by a sweep circuit which sequences through 30 or 120 energy steps. Essentially, the instrument package collects data on both the electrons and ions within the selected energy range concurrently in time but separately by charge polarity of the species. The triggering and timing operations of LEPA is done in conjunction with other systems.

At this point, it is sufficient to mention only briefly the sensors and systems that would contribute to the data gathering by LEPA. The onboard scientific magnetometer measures the geomagnetic field and notifies onboard systems of the directional angles of the field lines. The triggering of detection by LEPA for energetic particles within specific pitch angle ranges is governed by signals from the magnetometer. By observing the passing of the sun with a sensor, CRRES confirms its spinning rate with LEPA; the analyzers then synchronize their operation to execute an integral number of voltage sweeps per spin. Since the satellite is spinning, LEPA samples the region of space around the spin axis and needs only to store the data as a time-dependent series of sensors sweeps.

The spatial resolution of LEPA is delimited by its sensor field of view (FOV) subdivided into discrete angular bins. The FOV for the device is limited to an aperture width of 5.5 degrees with a elevation range from 26 to 154 degrees relative to the spin axis. This detection "fan" can angularly resolve the counts into sixteen 8-degree bins, arranged symmetrically with respect to the plane normal to the spin axis. These bins are henceforth referred to as "zones", and numbered 0 to 15. Zone 15 is covered and is used for collecting background counts. Each spin is divided into 64 sectors (5.6° per sector), with one voltage sweep executed per sector.

LEPA is designed to measure the flux at 30 or 120 energy steps for each of 64 sectors within each of 16 zones for ions and for electrons. Constraints on the telemetry data rate require reduction in the amount of data actually transmitted. To accomplish this, LEPA has a number of operational modes which sacrifice resolution in energy, direction, time, or some combination of these. The mode used at any given time is controlled by ground commands. The modes we are concerned with in this report are modes 0 and 10.

In mode 10 (sometimes designated "A") the voltage sweep sequences through 30 energy steps. The amount of telemetry data is reduced by combining the results (ion results are summed; electron results are averaged) for 4 adjacent sectors and two adjacent zones. The combined sectors and zones are sometimes called coarse sectors and zones. In mode 0, the ion data are collected in the same format as in mode 10. However, the electron data are collected only for sectors which contain the loss cone, or are located approximately perpendicular to the loss cone (1/4 spin from the loss cone), without being combined into coarse sectors or zones. There are usually 2 loss cone sectors and 2 "perpendicular" sectors per spin. Detection of these sectors is based on signals from the onboard magnetometer. Additional data are sent in these modes,

but not used in this work.

Regardless of the operating mode and voltage level, LEPA packs detector signals and passes the compressed data to the telemetry system. The transmitted data contains specific information assigned at particular locations within the byte streams that are related to the CRRES instrument packages and onboard systems, such as the sensors and magnetometer. These telemetry data are received, processed, and separated into files based on orbit number and a single-digit index related to placement on the CRRES body. The files of interest are the CMF####0 and CPC####0 files. The former data base contains the measurements made by the CRRES scientific magnetometer; the latter, the measurements made by various instrument packages, including LEPA. In this guide, the "####" field will represent the orbit number, which ranges from 63 to 1067 for LEPA. The suffix "0" in the file names means that those instruments are located on a specific portion of the CRRES body. Knowing the placement of the instruments is crucial when using attitude determination software to translate the look angles of the FOV originally defined in a satellite body-centered frame of reference to an ECI-based coordinate system.

The orbital files mentioned above are available to Phillips Laboratory staff personnel. Certain codes have been written to extract specific information necessary for post-processing. The CMF####0 files are processed directly by Radex to get ECI-based geomagnetic field vectors, and will be discussed in Section 4. The CPC###0 files have been processed by PL personnel to produce LEPA#### files containing only the LEPA data for orbit ####.

1.2 PROCESSING OF LEPA DATAFILES

LEPA#### is the instrument data base whose basic file format and procurement are detailed in Appendix B. The detailed contents of the data record depend on the operating mode, as described in *Hardy, et al.* [1993b]. To summarize the aforementioned document, LEPA detects background and incident counts as a function of zone and sector per channel for a given operating mode. Each data record contains measurements collected by LEPA within a half-spin.

LEPA software documentation [*Kerns*, 1993] describes how to extract the differential directional fluxes, time tags, look direction in the spacecraft coordinate system, and pitch angle. To perform this post-mission processing, PL staff personnel have written and used Borland Turbo PASCAL codes to strip out specific information from LEPA#### files. The PASCAL codes in question can read in a LEPA#### file, extract out specific details (dependent on the operating and voltage modes), and generate both the raw and averaged background counts. The embedded algorithms then test for consistent or corrupted counts. Portions of the code define calibration values and employ conversion expressions to assist in translating the net counts to equivalent differential directional flux values, $j(\Omega,E)$. Supporting routines are used to read post-processed magnetometer data and satellite ephemeris parameters concurrent to the time of the detector signals.

For portability and flexibility between machine platforms, Radex converted these routines over to FORTRAN and validated the translation by comparison between the outputs from the original and modified executables. The reason for the cross-language conversion is that, while the object-oriented features of Turbo PASCAL are useful, the compiler is available only on the DOS-based personal computer system. Conversion to FORTRAN enables a wider pool of platforms to be available for programmers and users to port codes, to modify existing modules, and to incorporate additional features within driver software. For example, LEPAFLUX is the simple driver using the converted routines to obtain the fluxes with the corresponding pitch angle for each sector and zone within every half-spin. LEPAFLUX then passes the information to a user-defined module where additional codes could be inserted. As another example, GET3DBIN uses these converted routines to derive and to bin individual differential directional fluxes prior to computing the mean differential directional fluxes separately for the electrons and ions.

2. THE BASICS OF THE LEPA FILE PROCESSING

Note that LEPAFLUX is significantly different from GET3DBIN in two aspects. First, LEPAFLUX reads the magnetometer files already created by PL staff. Second, the program does not presently have embedded routines for converting fluxes from a satellite body-centered coordinate system to other frames of reference. What is in common for both codes are the user-interface features, and use of general support routines designed for reading the LEPA data file to extract and convert the net counts into fluxes. For this reason, detailed discussion on LEPAFLUX is deferred to the next section after a brief mention in this section.

2.1 THEORETICAL MOTIVATIONS

It is necessary to establish an initial frame of reference that is satellite body-centered, i. e., the spacecraft coordinate system. As shown in Figure 1, the physical setup of CRRES is basically a rigid disk-like body bearing eight flat sides. Let there be an imaginary cylinder defined by the surface containing the flat sides on the satellite body. To complete the picture, let the "disk" rotate so that the spin axis is coaxially centered through the imaginary cylinder and exiting from the endcaps. This means that the eight flat sides move in a constant radius around the spin axis; the surface of the cylinder will then be rotating rather than the endcaps.

A normal projection from the spin axis passes perpendicularly through the flat plate where LEPA is mounted. This projection would be in the center of the FOV with eight zones on each side closer to the spin axis. By defining that normal projection via the FOV as the x-axis and the spin-axis, the z-axis, the y-axis nominally follows from applying the right-hand rule of orthogonality. Since LEPA is fixed relative to the satellite body, its FOV could only sweep in the positive x-z half-plane. The rotation of the satellite allows the x-axis to be spun about the z-axis. From the space-fixed frame of reference, this x-axis would be sweeping out an equatorial plane, while the look angles within the FOV fan would be able to scan up and down

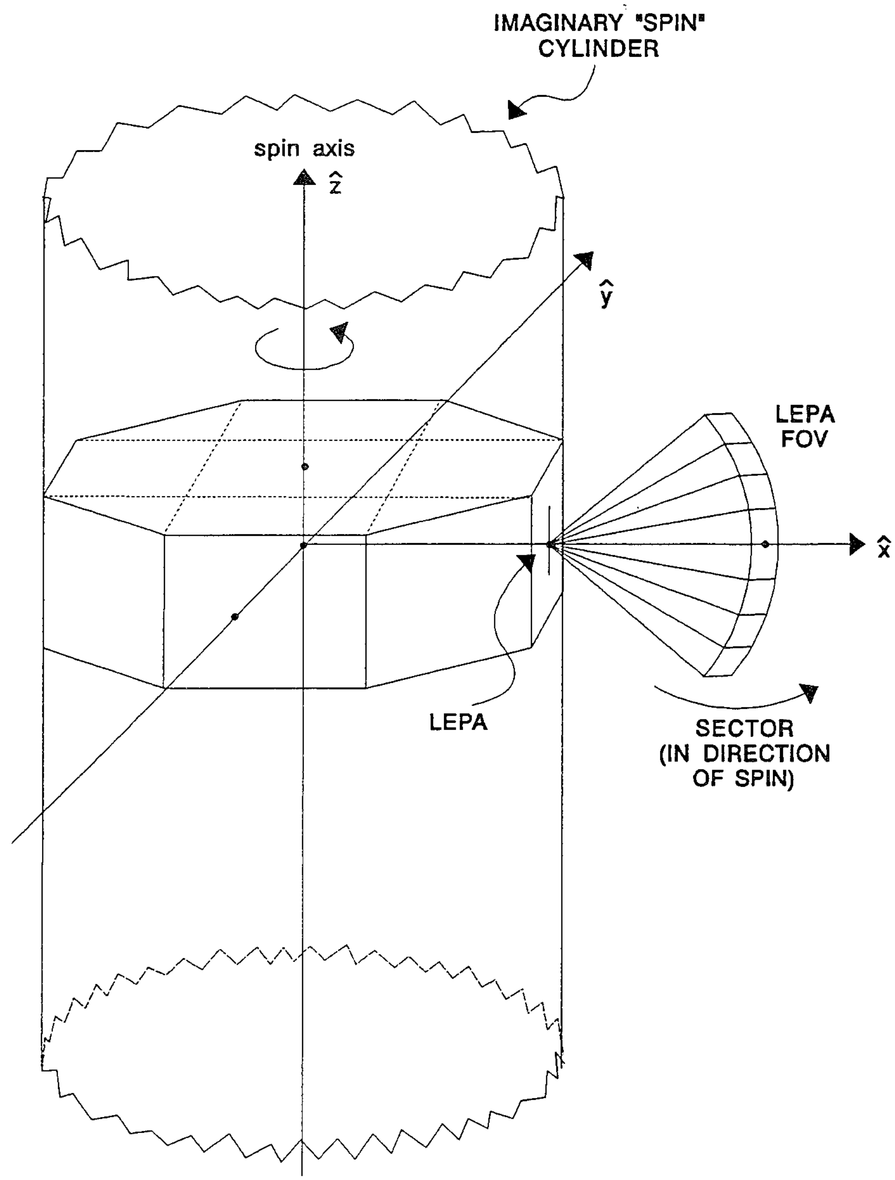

Figure 1. Generalized layout of the CRRES satellite body-centered coordinate system as defined for LEPA. Notice that the sensor FOV is fixed relative to the $\hat{z}-\hat{y}$ plane, but the spinning action of CRRES enables LEPA to sample angularly around the $\hat{z}$-axis.

through the plane. The discussion simply means that LEPA is capable of sampling around the spin axis of the satellite; the specific look angles due to a zone within the FOV fan will cover an solid annular ring parallel to the equatorial plane.

The raw counts as detected within a given sector and zone can be converted into directional differential number fluxes. The relationship between the raw counts and the resulting fluxes is given as

$$j(\Omega,E) = f_s\ \tilde{\eta}[\Omega]\ G[E,V]\ (n[\Omega] - \langle n[\Omega]\rangle_{bgnd})$$

where E is the energy; Ω is the FOV (more specifically, the zone and sector); f_s is the sweep frequency; G, geometric factor; n, detected raw count; $<n>$, averaged background count; and $\tilde{\eta}$, effective detector efficiency. The latter accounts both for zone-to-zone relative efficiency, and the fact that the coarse zones measurement are either added or averaged raw counts, depending on the species. See sections 4.3 and 4.4 of *Kerns* [1993] for more detail.

Both the LEPAFLUX and GET3DBIN codes can read the LEPA data file and compute the fluxes as discussed so far. At this point, LEPAFLUX will output the flux values, differentiated by species and operational modes, to a file before terminating its run; GET3DBIN does some additional processing with the data. GET3DBIN makes use of the known satellite attitude to transform the fluxes to a more suitable coordinate system. This is accomplished by knowing that the spinning rate of CRRES has been nearly constant, and that the orientation of the spin axis has already been worked out in earlier research. Therefore, any measurement made relative to the spin axis can then be transposed to a non-rotating coordinate system. All that is sufficient for LEPA operational goals is to make a series of measurements, time-tagged to the spin rate. Given the satellite spin rate, position, and attitude for all data points, coordinate transformations of the measurements are possible.

The current software uses the CRRES attitude determination routines, previously developed by Radex for coordinate transformation from a satellite body-centered frame of reference to a non-rotating system. For more details, see *McNeil* [1991]. Note that the technique allows transformation of both the measured geomagnetic field components and instrumental look directions to be mapped to the ECI frame of reference from the satellite body-centered coordinates. Once the ECI components are defined, standard coordinate transformation routines can be used to recast the values into another frame of reference for additional specialized post-processing.

In most studies of magnetospheric plasma, the plasma velocity distributions are considered to be gyrotropic, i. e., independent of the direction of the component normal to the magnetic field. The distribution function then depends only on the energy and the pitch angle, which can be determined entirely within the spacecraft coordinate system. To express meaningful results for non-gyrotropic distributions, it is necessary define a despun coordinate system, one not rotating with the satellite spin. It is further desired to have the z axis of this system aligned with the magnetic field, so that one can specify direction by pitch angle and azimuth, the latter being an

angle of rotation in the plane normal to the field.

To define the azimuth, one can invoke a rectangular coordinate system, called the "magnetic field meridian" (MFM) system, as follows:

z axis: parallel to the magnetic field direction B
x axis: parallel to the cross product (magnetic east) x B
y axis: completes right-handed rectangular system with z and x.

Magnetic east is the y or "D" direction in the "VDH" coordinate system (Fig. 3-20, from *Bhavnani and Vancour* [1991]), which has the dipole vector as its z, or "H" direction, while x ("V") points radially outward. If the magnetic field and the dipole vector are parallel, as is the case near the magnetic equator, then the coordinate system defined here is identical to the VDH system. One could define azimuth as being the angle of rotation from +x to +y.

Repeated sampling by or prolonged exposure to a detector is usually required in order to have confidence in the measured number of charged particles within a specific energy range. There are variations in background levels as well as particle density; spurious data points can and do occur. By nature of LEPA operations, data consist of only discrete components for the plasma energy spectrum. To address these issues in reconstructing the continuous energy spectrum in three-dimensional space, data from several spins must be combined or fitted.

In an attempt to combine the data on a physical basis, Radex has employed a binning scheme using the MFM coordinate system. By defining a two-dimensional grid based on azimuth and pitch angles, it is possible to map the measurements from an initially satellite body-centered frame of reference to, ultimately, the (Φ,α) points on the grid. The distribution of the discrete fluxes would hint at structures within the plasma data as detected by LEPA. By dividing the grid into intervals of azimuth and pitch angles as bins, several half-spins worth of measurements can be accumulated in order to increase confidence in the data above the background levels. Spatial resolution may be reduced but general trends can be determined and even fitted as a function of these indices.

The values for Φ are restricted to the interval 0 to 2π radians (360°); α, 0 to π radians (180°). Binning by Φ is simple; the range can be divided into equal-sized bins. Binning by α is not as simple, since higher resolution for extreme values is desired, to cover loss cone regions. To satisfy the requirement and yet provide equal-spaced intervals for mid-range, the end bins in α are made half the size of the other bins.

GET3DBIN uses the binning scheme to read in LEPA data for several half-spins, adding the fluxes into associated bins. Upon completion of a cycle of these half-spins, the code will determine the statistical information for each bin and output the information to a file. See Section 5.2 for details.

2.2 BASIC STEPS OF THE SOFTWARE PROCESSING

Except for the case of LEPAFLUX, the initial step is the extraction of ECI-based measured geomagnetic field components from the scientific magnetometer database. The processing ends when the three-dimensional distribution of the plasma density are computed for both the electrons and the ions. Note that the discussion on the pertinent major programs in each step is supplemented by more detailed sections later on in this report. The reader is encouraged to refer to the general operational flowchart in Figure 2 where the step by step procedure is depicted.

Starting with the data files related to the measurements made by LEPA and the scientific magnetometer, all files are processed in piecewise fashion. For each processing step, there are three major stages. The initial stage is to retrieve the necessary input files from various sources for each software package. The next stage is to create the output data files by running the code. The final stage is to validate the results by plotting the data. Each major code has a subsequent section devoted to it; within those sections are detailed information on how one can retrieve the input files, execute the main code, and post-process the output files.

Before proceeding with the discussion of software-related processing, an explanation is needed for the file name convention used in this manual. In the original convention, some of the file names are suffixed with '*####*0' where '*####*' is the orbit number of interest and '0' refers to the location of the instrument on the CRRES body frame. Files whose data are not dependent on instrument placement have been named as "ABCD*####*" where the first four letters tell what type of file it is. The latter convention has been used in the original PASCAL codes; the convention has been adopted in current codes to maintain consistency. The advantage of this renaming convention is that, when the user is forced to make a local copy with a slightly different name, this approach actually avoids accidental overwriting of the original files. In the codes, the actual filenames are hardwired as "ABCD*####*', where the CRRES body frame placement index can be inserted, if needed, into the "D" slot.

A programming discussion of these codes must be given first. All software has been written in FORTRAN language and compiled with Microsoft 5.1 FORTRAN compiler. By not incorporating any Microsoft WINDOW interface features, all codes are made to run on a basic DOS system under the 640 Kbyte RAM limits. Each code is designed to be portable to other systems as much as possible. There are three changes necessary to fully implement the software on any mainframe: the use of command-line argument utilities, bit manipulation routine calls, and one nonstandard library call. See "Appendix A: Porting To Other Systems" for additional details. For the enduser, it is sufficient to mention that most codes make use of either the *command line arguments* or a special user-modified input file called a *namelist* file, or both.

Command line arguments allow the user to enter additional arguments on the same line as the program name after the system prompt. Implementation of the command line arguments feature is system-specific, and is explained in more detail in Appendix A. The command line arguments are advantageous when the user just wants to do a single-line launch of the application code.

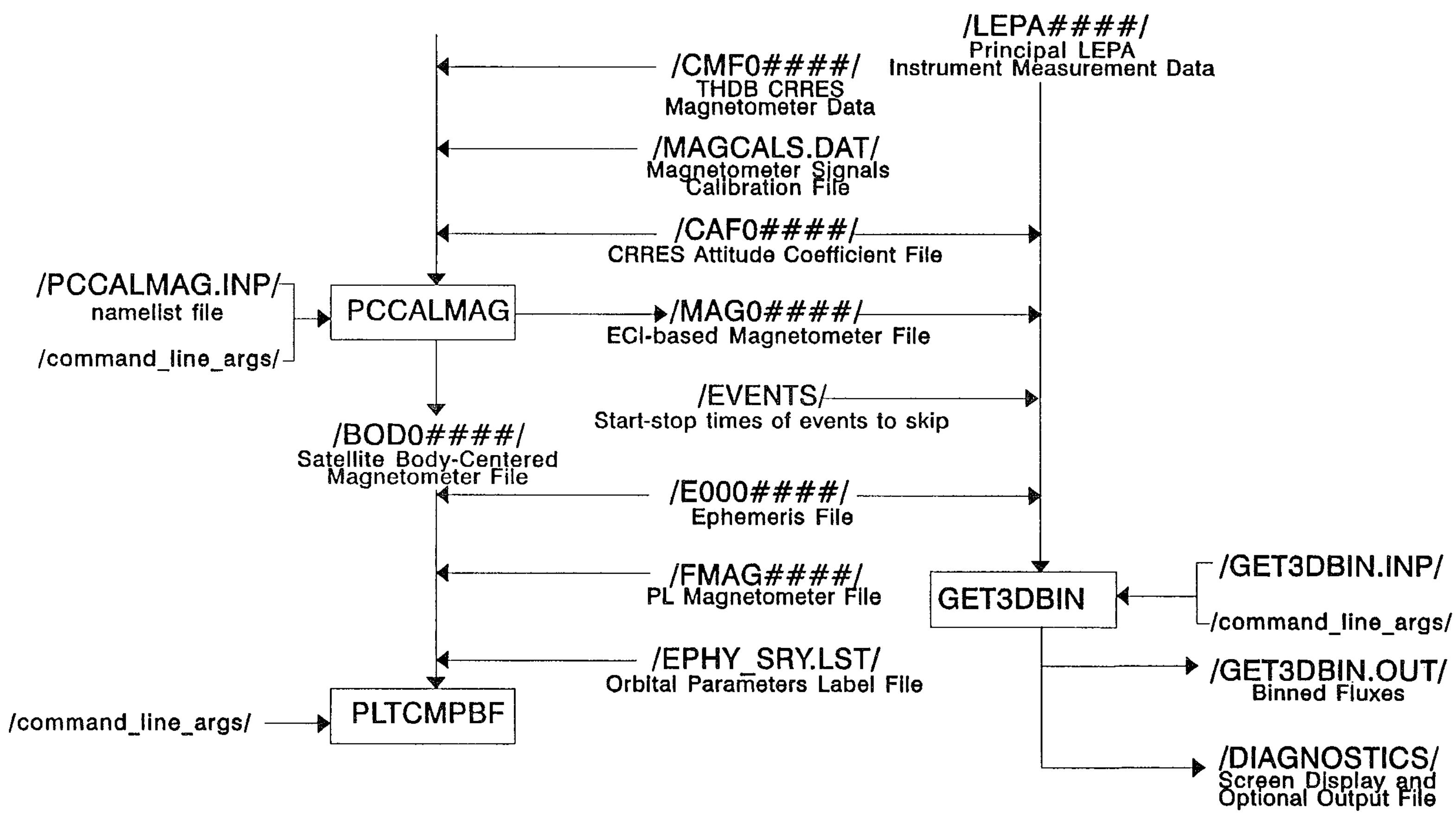

Figure 2. Basic components of the processing of LEPA data files as performed by various programs. The names of the major input/output files are given here for reference.

The feature is also suitable for use in a DOS batch file which could be used to run several orbits.

NAMELIST is an input/output utility that enables any program to read in a flexible number of values or to write out a diverse set of variables specific to a run. Although the NAMELIST feature is not in the FORTRAN-77 standard, it is available on most, if not all, compilers. The widespread implementation of NAMELIST by the majority of operating systems has encouraged its official recognition in the FORTRAN-90 standards. NAMELIST is typically useful when there is a large number of input parameters, most of which can be set to default values unique to the user. In such a case, the user need supply only those inputs which differ from the programmed defaults.

To run these codes, the typical procedure is to simply type "*program_name #### program_namelist_file*" to process the appropriate data files for the orbit numbered "*####*" using the input namelist file *program_namelist_file*. For convenience of the user and programmer, a namelist file has been provided to pass default parameters that could be changed from machine to machine, task to task. The default namelist file can be generated by simply typing "*program_name ####*". The program running for the first time will not find the default namelist in the current directory and will create one for the user before terminating. In regard to creating a default namelist, any legal orbit number actually can be used, since the program first looks for the file before processing any data.

Before running any software package, several input data files must be collected from various sources and several of them must be processed before use. Most of the files are from the CRRES historical database, which contains flight-time data from various onboard instrument packages, such as the magnetometer, and post-mission computed quantities, such as the satellite ephemeris. Of those files, some are available at Phillips Laboratory on-line via the mainframe accounts or on media not accessible by network systems. Retrieval of inaccessible data requires on-site visits once the user is given access authorization. Radex will provide the remainder of the necessary files. Instructions on how to acquire them will be specified in Appendix B.

All of the plotting packages mentioned in this report use GraphiC, a graphics utility package put out by Scientific Endeavors Corporation. Given the interfacing files necessary to work with Microsoft FORTRAN, the graphics library utility allows the user to first preview the plot on the screen and, if desired, create a hardcopy of the same plots on any printer. There are similarities between the types and levels of graphic routines calls to DISSPLA and GraphiC; this suggests that any conversion of the plotting code from DOS-based platform to the mainframe operating system would be straightforward. For additional details, see the appropriate graphic software manuals.

2.2.1 Magnetometer Database Processing

Magnetometer data files are necessary in order to compute the pitch angles for the directional differential fluxes. In the case of LEPAFLUX, there are already PL post-processed data files available for each orbit. See Section 3 for more details. GET3DBIN uses a magnetometer data file where the three components of the geomagnetic field are in an ECI reference frame. This section gives a brief summary of the process; actual instructions are provided in Section 4.

The first step to generate the suitable magnetometer datafile for GET3DBIN is running PCCALMAG. For input files, PCCALMAG uses the magnetometer THDB (Time History Data Base) and the calibration files. In addition, the CRRES attitude determination coefficient file is also required for processing. Appendix B will fully describe where and how to obtain those files. The only necessary discussion here is what the files represent. The magnetometer THDB is the telemetry database which contains signal strengths as measured by the scientific magnetometer. The calibration file is used for converting those sensor signals into an equivalent three-component geomagnetic field strength in a satellite body-centered coordinate system. The CRRES attitude determination coefficient file is the file necessary to transform any vector defined within the satellite body-centered system into the ECI-based coordinate system.

Running PCCALMAG actually requires two passes. The first pass is to generate the geomagnetic field components as measured in a satellite body-centered frame of reference for validation purposes. The second pass is to actually convert the magnetometer signals from the database into geomagnetic field components as would be seen in an ECI-based frame of reference at the satellite position. The purpose of the first pass is to validate that the processing of the magnetometer data base does not introduce any spurious or sudden discontinuity points in the output file. Probably the easiest and the most sensitive measure of anomalous behavior within the processed magnetic field is the abrupt change in the angle between the geomagnetic field vector and the spin axis of the CRRES satellite. This angle is referred to as the *elevation* angle, θ. A satellite body-centered quantity, the elevation angle is defined as

$$\cos(\theta_e) = \hat{z}_{sat} \cdot \hat{v}_{sat}$$

where the unit vector z aligns with the spin axis of the satellite, and unit vector v, with any vector measured in the satellite body-centered coordinate system. The elevation angle for the geomagnetic field vector is the easiest and quickest way to perform diagnostic studies on the newly-processed magnetometer file.

Thus, the validation process is generally quite simple, for it requires only that a graphical inspection of the elevation angle for the geomagnetic field be made in comparison to two other sources of geomagnetic field values. To accomplish this task of superimposing elevation angles from three different geomagnetic field-based data files in a plot, a plotting package called PLTCMPBF is used.

Described in Section 4.2, PLTCMPBF reads the newly-computed values and compares the secular trend and fine structure variations within the new data with those computed from two input files. One is the geomagnetic field model (IGRF-85 + Olson-Pfitzer) stored within the CRRES ephemeris file, E000####. The other, FMAG####, is 30-second interval field components previously processed by PL from the magnetometer THDB. In addition to the previously mentioned three files, PLTCMPBF also needs the CRRES attitude determination coefficient file to map the quiet geomagnetic model components from an ECI frame of reference to satellite body-centered coordinates.

Having established the input requirements, the validation procedure with PLTCMPBF actually consists of checking three features of the plot. First, the long-term trend of the measured geomagnetic field should follow the secular trend of the static model field for the quiet portion of the orbit. Second, the previously-processed magnetometer data file produced by PL staff should be closely followed, in the short-term behavior, by the newly-processed field. Finally, there should be a lack of abrupt discontinuities in the timeline plot of the processed data.

In the event that the magnetometer data passes the visual inspection, the second pass is then performed and the output magnetometer file is created, having the three components of the measured geomagnetic field defined in the ECI-based coordinate system. The newly-created file containing the ECI magnetic field vectors is available for the next step described in Section 2.2.2.

If the validation has revealed a possible problem with the processing of the magnetometer data file, modifications can be made to the several data editing algorithms within the code. See Section 4 for more details.

2.2.2 Binning of Differential Directional Flux

GET3DBIN is the next major code to run and requires several input files. More specific details are given in Section 5 on how to use/provide input values and select options. Here, it is sufficient to describe the steps necessary to generate the next level of output values. The primary input files are the LEPA data file and the newly-created magnetometer data file. The necessary additional input files are the attitude determination coefficient file and the ephemeris database already used in the previous phase.

The main functional role of this code is to compute the mean $j(\Omega,E)$ as a function of the (Φ,α) bin indices. The operational scheme is to first compute the net counts for each of the sectors and zones within the FOV for both the electron and ion constituents. The counts are converted into the instantaneous $j(\Omega,E)$. Once the value has been determined, the look angle for the sector and zone portion of the FOV is translated to the ECI-based coordinate system. By using the magnetometer data, the pitch angle is also computed. Given the magnetic field components, conversion from ECI to magnetic field meridian (MFM) coordinates is defined, and the azimuth angle follows from definitions based on the coordinate transformation as given in Section 2.1.

Each measurement is thus assigned an azimuth-pitch angle bin to go along with its association with a specific energy channel. Means and standard deviations of the fluxes over a specified number of half-spins are determined for each channel and directional bin. The results are saved to an output file whose format structure is described in Section 5.2.

3. OPERATION OF PROGRAM LEPAFLUX

This section describes the program LEPAFLUX initially in terms of overall functional role with regard to LEPA datafiles. The discussion centers on basic operational features of the software package. The first subsection will explain how and what input/output files are needed.

LEPAFLUX is a PC-based software utility that reads in a LEPA datafile to find the pitch angle (α) and corresponding differential directional fluxes, $j(\Omega,E)$, of the ion and electron distributions. Basically, LEPAFLUX extracts raw and averaged background counts from measurements made by LEPA within a specific zone and sector for each energy channel, and converts the net counts into $j(\Omega,E)$. The fluxes are stored by energy channel number and voltage level for each sector and zone. The pitch angles are computed and stored either as its cosine or angular value measured in radians or degrees. Depending on the user's choice, the printout of the stored values can be made for every $N_{1/2}$ half-spins. There is no coordinate transformation performed; all values are given in the spacecraft coordinate systems.

The chief feature of this code is the inclusion of a subroutine, "USER_MODULE". As is currently designed, "USER_MODULE" just prints out the stored values of α and $j(\Omega,E)$ for every $N_{1/2}$ half-spin. Here is where a user could implement additional post-processing schemes for the collected fluxes and corresponding pitch angles. The advantage is to provide a "hook" without any necessary modifications to the supporting routines.

LEPAFLUX reads both the command line arguments and a NAMELIST file for orbit-dependent and input file-specific values. The user is required to enter the orbit number on command line, followed (optionally) by the name of the namelist file, which overrides the default name LEPAFLUX.INP. The namelist file contains the pathnames necessary to find the required input files to do a run. Also within the namelist file is a switch for controlling the range of data analysis.

To process a LEPA file, LEPAFLUX needs two other orbit-dependent datafiles. They are used to compute the time-dependent pitch angle, and to obtain ephemeris-dependent quantities. These files are available at Phillips Laboratory. The first file is the geomagnetic components measured by the onboard CRRES magnetometer in the satellite body-centered coordinate system. The magnetometer data file had been processed by the original PL code, CMF2FMAG, and is named "FMAG####". The second file, E000####, the ephemeris file, contains crucial values, such as satellite position and velocity, sun look angle, and geomagnetic model values. The latter file is not actually used, but is provided in case the user wishes to use the information for additional

post-processing of the fluxes.

3.1 OVERALL GENERAL DESCRIPTIONS FOR INPUT

The necessary user-dependent inputs are given both as command line arguments and from within a NAMELIST file. On the command line, the user enters the orbit number and the name of a NAMELIST file if different from "LEPAFLUX.INP". This latter feature provides flexibility of using different NAMELIST files tailor-fit for special cases, such as different LEPA operational modes. The NAMELIST file provides file pathnames, assigns data control parameters, and sets an optional switch. The NAMELIST options and command line arguments will be described later.

To support certain libraries within the code to compute various quantities, there are three supporting data files necessary for execution of the code. The paragraphs immediately following this portion deal with command line and NAMELIST arguments. Following the discussion on direct input, an explanation will be given on how to obtain and process the appropriate input datafiles.

3.1.1 Command Line Descriptions

To run LEPAFLUX, at the DOS prompt, enter the command line with options:

LEPAFLUX *####* [NAMELIST_file]

where

####	==	orbit number in integers (63-1067) and
[NAMELIST file]	==	path and name of NAMELIST file if NOT "LEPAFLUX.INP".

Note that, in the description of the command line arguments in this discussion, the brackets ("[ ]") are not actually typed by the user, but serve as a conventional means of conveying the fact that the enclosed argument is optional.

The orbit number MUST be the first argument and the NAMELIST file the second. By default, the NAMELIST file is assumed to be in same directory as LEPAFLUX. If the NAMELIST file is located within a different directory, the filename to be entered on the command line must have a full path properly prefixed to the name.

LEPAFLUX terminates if the first parameter is not a legal orbit number, or if the NAMELIST file does not exist. In the latter case, LEPAFLUX first graciously provides the user with a

complimentary NAMELIST file named and located as requested.

3.1.2 Namelist File Descriptions

The NAMELIST file contains user-defined choices in file management, binning limits, and option switches. The file management aspect deals primarily with where the input files are located, what names are assigned to the output files, and where they are to be placed. Limits on data collection are done for either ions or electrons, or both. These limits control the cutoff for the channel index, or disabling the use of data from coarse zone 0. Table 1 gives the list of namelist variables with associated comments and states whether the value is required or optional.

TABLE 1. Namelist Variables Used as Input in "LEPAFLUX"

NAMELIST VARIABLES:	REQUIRED OR OPTIONAL	COMMENTS
LEPA_PATH	REQUIRED	
FMAG_PATH	REQUIRED	
EPHM_PATH	REQUIRED	
TEMP_PATH	REQUIRED	
NO_OF_HALF_SPINS	REQUIRED	
MAX_ION_CHANNEL	REQUIRED	MAXIMUM VALUE = 28
MAX_ELECTRON_CHANNEL	REQUIRED	MAXIMUM VALUE = 19
FILE_OUTPUT	REQUIRED	
NO_COARSE_ZONE_ZERO	OPTIONAL	RECOMMENDED "TRUE"

The following is an example of the default namelist generated on DOS:

```
&INPUT_DECK
LEPA_PATH = 'D:\CRRES\LEPA_LIB\LEPA0000\'
FMAG_PATH = 'D:\CRRES\FMAG\'
EPHM_PATH = 'D:\CRRES\EPHM\'
TEMP_PATH = 'F:\'
NO_OF_HALF_SPINS = 1
MAX_ION_CHANNEL = 28
MAX_ELECTRON_CHANNEL = 19
FILE_OUTPUT = 'E:\LEPAFLUX.OUT'
NO_COARSE_ZONE_ZERO =  T
/
```

The first line is the name of the namelist deck. The last line, a simple "/", is the namelist terminator. The values of "T" or "F" are actually "TRUE" or "FALSE" as written out by Microsoft FORTRAN. The fields for "?_PATH" are file prefixes that tell LEAPFLUX where the necessary files are located. The "TEMP" path is needed to store special interim files that are not needed once the run is done. The TEMP_PATH is usually pointing to a DOS RAMDISK to improve speed. These files are discussed in Appendix B.

The NO_OF_HALF_SPINS is the number of records which the program will process before it updates the output file. In the default USER_MODULE supplied by Radex, this is the number of half spins which are skipped, if greater than 1, before data are output. If the user incorporates any summation scheme and must specify the number of half-spins ($N_{1/2}$) over which the fluxes are to be added up, the author recommends that an even number be used to allow symmetric sampling of plasma distribution. Operationally, setting $N_{1/2}$ to 4 will yield timesteps spaced a minute apart. For a typical orbit whose period runs nearly ten hours, this means about 600 time-tagged points are printed. The actual number of points may decrease if there are gaps due to data dropout, mission events, or corrupt counts detected in the file.

The code is set up to sort the electrons and ions separately by channel number; there is a feature to allow for a different number of maximum channel numbers to be used. These adjustable parameters are more important than one may realize. The maximum energy channel numbers relate to the minimum energies that LEPA can detect (channel number varies inversely with energy). The absolute maximum channel numbers, 19 for electrons and 28 for ions, correspond to the minimum energies 100eV and 10eV, respectively.

The principal output file is "FILE_OUTPUT". The file contains the information on what is stored in specific arrays for the requested number of half-spins with a time tag. See section 3.2 for details on the data structure of the output file.

A logical switch, "NO_COARSE_ZONE_ZERO", is a flag to turn off reading information from the coarse zone zero region of the FOV field. This is to account for the fact that half of the coarse zone seven cannot be used due to poor detection efficiency. By forcing the program to consider only coarse zone 1 to 6, a symmetric spread of data can be realized within the LEPA field of view.

3.1.3 Input Files

Before running LEPAFLUX, three input files are needed. The primary input file is LEPA*####*. FMAG*####* provides the magnetometer data. E000*####* supplies the CRRES ephemeris and geophysical data related to the satellite positions.

3.2 OVERALL GENERAL DESCRIPTIONS FOR OUTPUT

The output of LEPAFLUX consists of sequential screen dumps and one primary output file, named "LEPAFLUX.OUT" by default. The primary output file contains, for both the ion and electron species, their differential directional fluxes and corresponding pitch angles for a given set of $N_{1/2}$ half-spins sampled from the LEPA datafile. The following subsections will describe in more detail the output format and explain what or how to interpret the values.

3.2.1 Screen Dump

At the beginning of the run, LEPAFLUX displays the header information for each file successfully opened within the initialization portion of the code. The first file to be opened is the LEPA file whose header information is written out as one item per line with label. The second file opened is the FMAG file, where the starting and ending time of available data in the magnetometer file are written out as UT in milliseconds. The same is true for the E000 file, the third file to be opened, save that its number of records read in are also written out.

Currently, LEPAFLUX will print out the UT for the end of each $N_{1/2}$ half-spins in both the millisecond and HH:MM:SS.SSS formats. The program will also print out the number of steps completed for every set of $N_{1/2}$ half-spins. The programmer can modify the USER_MODULE to also print out appropriate messages to the screen for any additional post-processing results.

3.2.2 Structure and Format of the Primary Output Data File

LEPAFLUX creates one primary output data file whose default name is "LEPAFLUX.OUT". It is an unformatted datafile with one header record followed by sequential data records containing the α and $j(\Omega,E)$ values. Table 2 gives the structure of the header record with the format necessary to read it. The subsequent data records contain the fluxes and pitch angles for all viable channel numbers as a function of voltage level, operational mode, species type, and sector/zone indices.

TABLE 2. Header Variables for Main Output File

HEADER VARIABLES	UNITS	TYPE
ORBIT NUMBER	63-1067	INTEGER*2
YEAR	90 OR 91	INTEGER*2
DAY OF YEAR	1-365	INTEGER*2
STARTING TIME OF ORBIT	UT, milliseconds	INTEGER*4
ENDING TIME OF ORBIT	UT, milliseconds	INTEGER*4
MAXIMUM ELECTRON CHANNEL NUMBER	0-19	INTEGER*4
MAXIMUM ION CHANNEL NUMBER	0-28	INTEGER*4

Since the current programs are capable of reading in only the mode 0 and mode 10 data from the LEPA file, there are four possible types of data that could be written to the primary output file. The types are electron mode 0 loss-cone sector, electron mode 0 perpendicular sector, electron mode 10, and ion mode 0/10. The first two types only have data from one sector, but 14 zones; the latter two, from eight sectors and 6 or 7 zones. This means that each type requires a slightly different way of presenting the range of fluxes and pitch angles possible.

To minimize the extensive coding in order to write and read the data in the output file, each data type is required to use three distinctive records. The first WRITE creates a record of four variables associated with the data type, the next gives the pitch angles, and the last, the fluxes.

For the first record, there are four variables uniquely associated with the data type. The first variable is a five-byte character label signifying the data type. The second and third variables are 4-byte integers indicating the number of sectors and zones, respectively. The last is a 4-byte integer, giving the UT time in milliseconds, either as a scalar or an array of 8 elements. The 5-byte label for mode 0 electron cases are 'LOSSC' and '90DEG'; mode 10 electrons and ions, 'ECRSE' and 'IONS '. The first record is designed to handle all four of these types.

Using the FORTRAN convention, the first record can be written as

" WRITE(*ifile*) *data_type*, N_{sector}, N_{zone},(*UT*(i), i=0, N_{sector})"

where N_{sector} is set to zero for electron mode 0, and 7 for others. Then, using the similar FORTRAN convention, the next two write statements corresponding to mode 0 data can be thought of as:

" WRITE(*ifile*) ((α(iz, iv), iz=1, N_{zone}), iv= *low_volt, high_volt*)"
" WRITE(*ifile*) (((*j*(ic, iz, iv), ic=0, $N_{channel\_type}$), iz=0, N_{zone}), iv= *low_volt, high_volt*)"

and for mode 10:

" WRITE(*ifile*) (((α(iz, is, iv), iz=0, N_{zone}), is=0, N_{sector}), iv= *low_volt, high_volt*)"
" WRITE(*ifile*)((((*j*(ic, iz, is, iv), ic=0, $N_{channel\_type}$), iz=0, N_{zone}), is=0, N_{sector}), iv= *low_volt, high_volt*)"

where the two 4-byte integer parameters, *low_volt* (set to 0) and *high_volt* (set to 1), are the limits in the implied do-loop for reading in voltage levels. The 4-byte real array for the pitch angles is represented by α; the 4-byte real array for the fluxes, *j*. The looping variable for sector-based data is *is*; for zone-based data, *iz*. $N_{channel\_type}$ is the maximum channel number specific either for ion or electron measurement.

Two special forms of the first data record are used to denote the end of the current half-spin case and end of file. The former is written as

" WRITE(*ifile*) 'ENDHS', 0, 0, *UT_AT_END_OF_HALF-SPIN*"

and the latter,

" WRITE(*ifile*) 'EOFLE', 0, 0, 0".

The way LEPAFLUX is designed, if certain mode-type data is never present for that $N_{1/2}$ half-spin case for either the ions or electrons, no zero-filled arrays will be written out. This minimizes the size of the output file. Due to the complexity of this file, a sample source code, to assist the user, is given in Appendix C.

4. PROCESSING OF THE MAGNETOMETER DATABASE

GET3DBIN uses the measured geomagnetic field components in an ECI-based coordinate system to compute and bin $j(\Omega,E)$ values. The code to provide the requisite magnetometer data file is PCCALMAG. A DOS-based utility, PCCALMAG is designed to process the historical magnetometer data imported from the PL mainframe and creates a file with either satellite body-centered or ECI-based geomagnetic field vector components as measured by the science magnetometer. It employs extensive data editing based on mission events and data outliers. By default, it generates the file in a form suitable for data validation with other sources of the geomagnetic field.

To perform a validation check, the geomagnetic field elevation angle is primarily used. Experience has been that elevation angle for the geomagnetic field vector is the easiest and quickest way to perform diagnostic studies on the newly-processed magnetometer file. Using elevation angles from three different geomagnetic field-based data files, validation is done with a plotting package called PLTCMPBF.

PLTCMPBF, short for PLoTting the CoMParison of B Fields, shows three overlapping curves representing the elevation angles of the geomagnetic fields derived from the three different files. Essentially, the program first reads a CRRES ephemeris file to extract the ECI-based quiet model geomagnetic field components, and uses the attitude determination coefficient file to convert the ECI-based coordinates to body-centered values before calculating the elevation angle. Next, the elevation angles are read directly from the FMAG file. Finally, the body-centered version of the magnetometer file generated by PCCALMAG is processed to obtain the elevation angles.

Section 4.1 gives the essential details of how to set up a run for PCCALMAG, and what the code does internally. Section 4.2 covers the operation of PLTCMPBF in similar details.

4.1 PROCESSING THE MAGNETOMETER DATA FILE WITH PCCALMAG

The measured geomagnetic field components in an ECI-based coordinate system are needed in order to compute and bin $j(\Omega,E)$ values within GET3DBIN. The code necessary to process the magnetometer data base and do the coordinate transformation is PCCALMAG. Its routines are derived from an earlier FORTRAN code written on VAX, CALMAG [*McNeil*, 1993]. Using algorithms approved by the Principal Investigator of the CRRES science magnetometer, CALMAG works with data which were sampled 16 times per second. The original software package uses an input calibration file to convert the three-axis fluxgate magnetometer signals into equivalent geomagnetic vector field components as measured in the satellite frame of reference. The software incorporates a data editing scheme to check for and eliminate known types of major spurious signals.

PCCALMAG is essentially an upgraded version of the original code and designed to run under DOS. Written to take full advantage of the command line arguments and a NAMELIST file, it can process the official binary Time History Data Base (THDB) formatted files. Some of the modifications allow for the fact that, within the THDB files, the magnetic field is sampled 8 times per second. Other modifications incorporate additional data editing features, such as ignoring data for LASSII mode and other undesired mission events. Optionally, it also transforms the geomagnetic field components from the satellite body-centered to the ECI-based coordinate system, using the CRRES attitude determination software routines.

4.1.1 Input to PCCALMAG

The code requires both the command line arguments and namelist file for user-specific inputs. The command line options enable the user to specify the orbit number, filename of the NAMELIST file (if not named as "PCCALMAG.INP"), choice of coordinate systems, and switches for no-bell and debug mode. The namelist file will allow the user to specify the path names of the files. The primary input file is the magnetometer THDB, with additional input data files used for calibration parameters and attitude determination of the satellite. The default output file is the time-tagged listing of geomagnetic field vector components, as measured in the satellite body-centered coordinate system. By selecting a command line option, the vector quantities are translated into the ECI-based coordinate system.

4.1.1.1 *Command line arguments for PCCALMAG*

To run PCCALMAG, at the DOS prompt, enter the command line with options:

PCCALMAG *####* [NAMELIST_file] [-ECI] [-NOBELL] [-DEBUG [UTSEC_MIN] [UTSEC_MAX]]

where

####	==	orbit number in integers (63-1067);
[NAMELIST file]	==	path and name of NAMELIST file if NOT "PCCALMAG.INP";
[-ECI]	==	switch for controlling the desired coordinate system;
[-NOBELL]	==	turns off the "bell" sounds upon completion of run;
[-DEBUG]	==	enables printing out selected data points;

[UTSEC_MIN UTSEC_MAX] == orbital time in seconds if debug is selected.

The orbit number MUST be the first argument and the NAMELIST file the second. By default, the NAMELIST file is assumed to be in same directory as PCCALMAG. If the NAMELIST file is located within a different directory, even with the name "PCCALMAG.INP", the filename to be entered on the command line must have a full path properly prefixed to the name.

The "ECI" switch controls the coordinate system of the output file. Without this switch listed on the command line, the geomagnetic field vector components are left in the satellite body-centered frame of reference and written out to the output file named "BOD0####.DAT". If the switch is enabled, the vectors are translated from satellite body-centered coordinates to ECI-based coordinates and written to the output file named "MAG0####.DAT".

"NOBELL" switch is for switching off audible signals. Nominally, the code beeps on completion of the run or runstop error. This allows the user to work nearby while the program processes the magnetometer data file.

The "DEBUG" switch is used for printing out the individual data points that pass the majority of data editing schemes. This feature is useful for printing out to the console (or when redirecting to file) the signals, low/high states, and other information that may require additional editing logic to be incorporated into the code.

If DEBUG is selected and no numbers are entered, then the code assumes that the entire orbit is to be printed out. A feature has been added to read in the lower and upper values of UT in seconds (minimum is 0, maximum is 122400 seconds) so that only the selected subsection of the orbit could be printed out.

4.1.1.2 *Namelist input file for PCCALMAG*

The default name for the namelist file is 'PCCALMAG.INP'. The NAMELIST file contains the full names (including path) of the calibrations file, other input files, and the output file. The following is the default NAMELIST file as generated under DOS:

```
&INPUT_DECK
MAGCALS_DATABASE = 'D:\CRRES\MAGCALS\MAGCALS.DAT'
CMF0_PATH = 'E:\CRRESMAG\'
BOD0_PATH = 'D:\CRRES\BOD0\'
MAG0_PATH = 'D:\CRRES\MAG0\'
CAF0_PATH = 'D:\CRRES\CAF0\'
/
```

4.1.1.3 *Input data files for PCCALMAG*

The main input data file for this code is CMF0####. It is the local file version of the Magnetometer Time History Data Base File. MAGCALS.DAT is an ASCII datafile representing the Magnetometer Calibration Data Base, and it contains, for most orbits, a set of calibration parameters used for converting sensor signals to equivalent physical units. The CAF0#### file

is needed for two reasons: 1) to turn off the data averaging when not in normal mission mode (canister ejection, attitude adjustment, spin changes, etc.); and 2) to convert from body-centered coordinates to ECI-based coordinates on the fly and bypass the need to save the body-centered values.

4.1.2 Output File from PCCALMAG

The output file is named 'BOD0####.DAT' or 'MAG0####.DAT', depending on whether the user desires a body-fixed or an ECI-based coordinate system. The structure of both ASCII data files consists of a single header record containing integers, separated by spaces, denoting the orbit number, modified Julian date, year (90 or 91) and day of year (1-365) for the start time of the orbit, and the orbit number from the calibration file that actually provides the signal conversion coefficients. The subsequent data records have four real*8 values as listed in Table 3 for the body-centered file:

TABLE 3. BOD0#### Data Record Variables and Format Structure

VARIABLES	UNITS	FORMAT
TIME TAG	UT, seconds	1F14.3
B_x	nT	1F10.1
B_y	nT	1F10.1
B_z	nT	1F10.1

and Table 4 for the ECI-based file:

TABLE 4. MAG0#### Data Record Variables and Format Structure

VARIABLES	UNITS	FORMAT
TIME TAG	UT, seconds	1F14.3
B_x	ECI, nT	1F11.2
B_y	ECI, nT	1F11.2
B_z	ECI, nT	1F11.2

Note that the format already includes implied spaces. This means that any program reading the file would use "FORMAT(1X,1F14.3,3F10.1)" for the former file and "FORMAT(1X,1F14.3,3F11.2)" for the latter.

4.1.3 Magnetometer Data Editing in PCCALMAG

PCCALMAG takes the eight-sample per 1.024 seconds snapshot of each geomagnetic field component and finds the mean value and corresponding weighted time tag. If there are one or more bad samples within the interval, the code excludes them from the averaging process. If there are no acceptable data points within the interval, the program skips to the next data record. Usually, the magnetometer data is spaced nearly a second apart, except for occasional sporadic intervals of bad or meaningless data.

PCCALMAG edits the data from CMF0#### file for two sources of errors: spurious magnetometer measurements and mission events that could lead to incorrect attitude determination. The first type of editing attempts to handle discontinuities or other electronic errors. The second type relates to satellite maneuvers or other events that cast greater uncertainty in attitude determination or fine-scale resolution of pitch angle.

The magnetometer data consist of three axial signals time-tagged within the telemetry data stream. Their values are to be limited to specific ranges defined by internal constraints. Values outside these ranges are signals of missing or invalid data, and are therefore not processed further. There are sometimes unacceptable values for one component or more for a short time interval. PCCALMAG looks for such unacceptable values and excludes them from processing. There are times when the data have a short time interval of non-physical results for various reasons; PCCALMAG skips over those intervals. Sometimes, the signals have a large change in one or more axial signals that does not exceed the limits. Usually, the data would drop or rise in one component while the others do not have this effect. To guard against this problem, PCCALMAG compares each component against a scaled range of previous immediate values; the scaling has been adjusted to allow for onset of "storm" events, yet filter out the spurious large abrupt discontinuities.

Any remaining spurious data points usually occur because a large discontinuity is still present within a scaled range for one component of the magnetometer measurement. To correct this problem, the editing needs to be a bit more restrictive in the scaling range. The user can adjust the scaling constants internally; the caveat is that the user must allow some leeway in data fluctuations due to magnetic "storms" effects or other related events.

The other undesirable portions of the data are when CRRES performs mission events or enters mission modes leading to indeterminant attitude or magnetic field orientations. The examples of these events are CRRES entering the LASSII mode portion of the orbit, crossing the earthlimb shadow, or ejecting canisters. Whenever there are attitude adjustments, they introduce rapid short-term periodic components to the magnetic field orientations. Since GET3DBIN uses a 4-point polynomial fitting function to interpolate between adjacent time points for the geomagnetic components, incorrect determination of the magnetic field components will result. For this reason, PCCALMAG also excludes these intervals.

4.2 VALIDATION OF THE MAGNETOMETER DATA FILE WITH PLTCMPBF

Using elevation angles from three different geomagnetic field-based data files, validation is done with an interactive plotting package called PLTCMPBF.

The verification check is accomplished by comparing the values of the elevation angles for the geomagnetic fields from two other sources with those of the interim data file. The comparison consists of three parts. First, the user must compare the long-term trend exhibited in the values derived by PCCALMAG with that of the values for the quiet geomagnetic field model and to inspect the newly-computed values with those of the FMAG file.

As it turns out, the easiest and most suitable means of verification is to plot the time history of the elevation angles for each of the magnetic field files. That is, the angle between the CRRES spin-axis and the magnetic field provides the most sensitive and global measurement of the geomagnetic field quantities while allowing comparison of trends with the computed quiet model.

Using the GraphiC 6.1 routines with the MS FORTRAN 5.1 compiler, PLTCMPBF makes a single plot with three curves. First, the program reads a E000*####* file to extract the ECI-based quiet IGRF-85 model geomagnetic field components and converts them to body-centered values using the attitude-correction CAF utilities before calculating the elevation angle. Next, the elevation angles are read in from the FMAG file. Finally, the BOD0*###* file generated by PCCALMAG is processed for elevation angles.

4.2.1 Input to PLTCMPBF

As is done in PCCALMAG, PLTCMPBF is written to take full advantage of the command line arguments and namelist file. The command line options enable the user to specify the orbit number, filename of the namelist file, if not named as "PLTCMPBF.INP", choice of starting and ending times within the specified orbit, as well as the debug mode. The namelist file will allow the user to specify the path names of the input and output files. There are several data files that are needed for running this plotting package.

4.2.1.1 *Command line input to PLTCMPBF*

To run PLTCMPBF, at the DOS prompt, enter the command line with options:

PLTCMPBF *####* [namelist_file] [-CUT [UTSEC_MIN] [UTSEC_MAX]]

where

== orbit number in integers (63-1067);

[namelist file] == path and name of namelist file if NOT "PLTCMPBF.INP";

-CUT == used for plotting and printing out the individual data points within the time slot bracketed by the passed UT times.

If CUT is selected and no numbers are entered, then the code assumes that the entire orbit is to be printed out. Otherwise, the lower and upper values of UT in seconds (minimum is 0, maximum is 122400 seconds) are given as integers to select a subsection of the orbit. For example, if the user suspects that a stray bad point from the PCCALMAG editing slipped through, a "cut" view of the orbit can be "zoomed", and the trend can be seen visually. Meanwhile, the code automatically saves any computed value to a "PLTCMPBF.DMP" file with the quantity UT (in milliseconds) and the elevation angles.

4.2.1.2 *Namelist input file to PLTCMPBF*

The default name for the namelist file is 'PLTCMPBF.INP'. The NAMELIST file contains the full names (including path) of the calibrations file, other input files, and the output file. The following is the default NAMELIST file as generated under DOS:

```
&INPUT_DECK
EPH_SRY_DATABASE = 'D:\CRRES\EPHM\EPH_SRY.LST'
BODO_PATH = 'D:\CRRES\MAGO\'
CAFO_PATH = 'D:\CRRES\CAFO\'
EPHM_PATH = 'D:\CRRES\EPHM\'
FMAG_PATH = 'D:\CRRES\FMAG\'
BODO_DOT =          16
EPHM_DOT =           0
FMAG_DOT =           3
/
```

The first variable is the full filename of the 'EPH_SRY.LST' data file; the next four are the filepaths to the other data input files. The last three designate the symbol desired for the three corresponding curves as defined by the GraphiC plotting utilities. The default values define a small solid dot for the output data from PCCALMAG, an empty square for PL magnetometer data, and a "+" for the model value from the ephemeris file. See the graphics package manual for more details about the choice of symbols.

4.2.1.3 *Input data files for PLTCMPBF*

The input files needed are the binary PL files 'FMAG####' and 'E000####'; the ASCII Radex files, 'CAF0####' (renamed from 'CAF####0') and the PCCALMAG file, 'BOD0####.DAT'. Also needed is a special ASCII database provided by PL staff called 'EPH_SRY.LST', which contains the information about the time and dates for each orbit. The code uses this file for time and date labels of the orbit.

4.2.2 Output from PLTCMPBF

PLTCMPBF initially generates a screen graphics plot as seen in Figure 3. The plot in turn can be saved to a specific graphics format file or can be sent directly to a hardcopy device. More

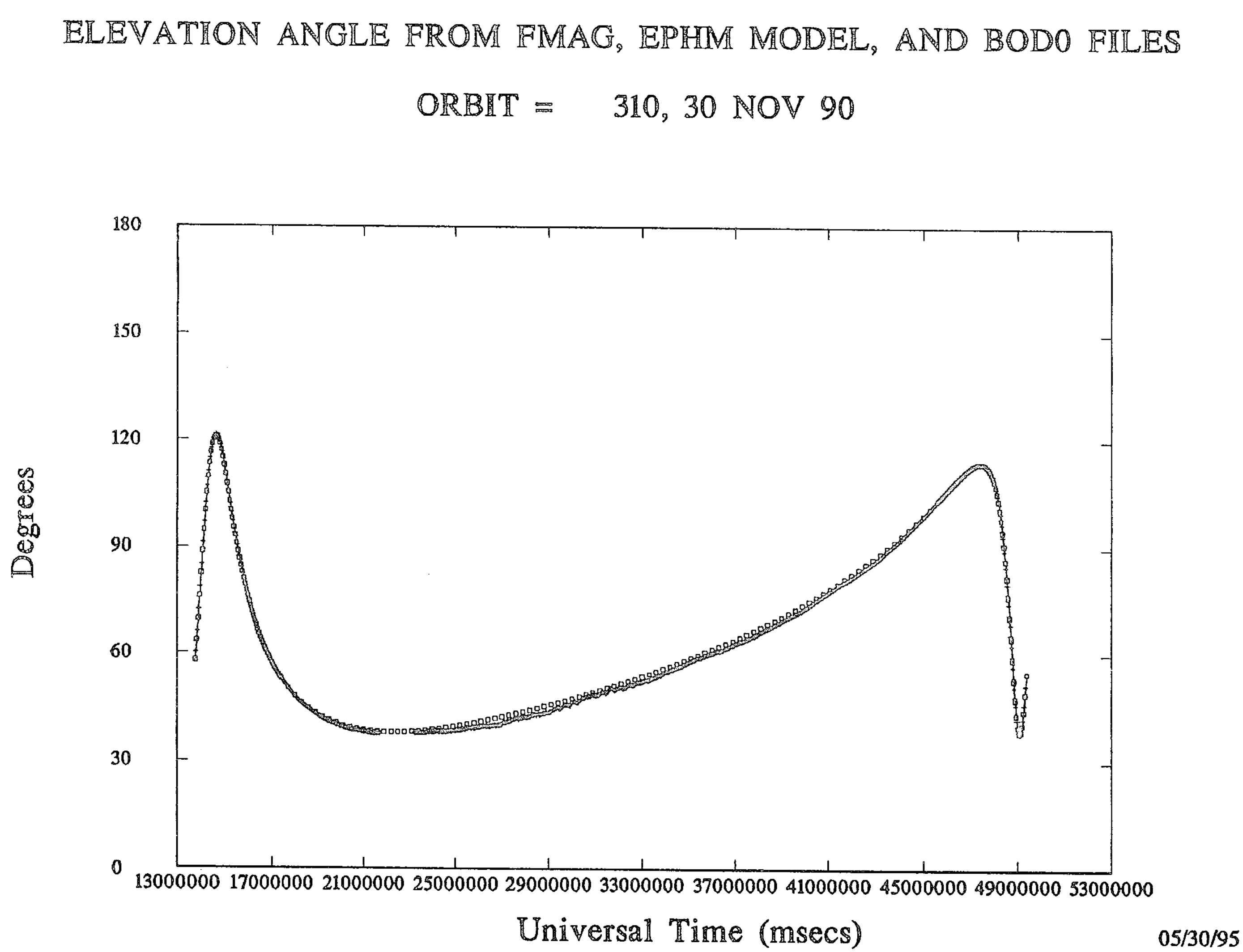

Figure 3. A sample plot of the elevation angle for orbit 310 from three different data files.

details on choices available in generating these outputs are available in GraphiC manual.

If the command line option "CUT" is chosen, a complimentary output file "PLTCMPBF.DMP" is generated as described in 4.2.1.1, containing the UT time and the three elevation angles. This output serves to help troubleshoot the PCCALMAG routines in its data editing as described in 4.1.3.

5. OPERATION OF PROGRAM GET3DBIN

GET3DBIN is a PC-based software utility that reads in a LEPA datafile to calculate user-defined interim quantities and mean differential directional flux values of the ion and electron distributions. More specifically, GET3DBIN first scans a specific number of half-spin data records to extract net counts and converts them into differential directional flux distributions as functions of energy and look angles relative to the CRRES spacecraft frame of reference. The code then does a coordinate transformation from the spacecraft frame of reference to the magnetic field meridian frame, in order to sort the measurements by pitch angles (α) and azimuth (Φ). Upon completion of reading in a user-defined number of half-spin records, GET3DBIN performs basic statistical analysis on all of the non-zero populated bins, and saves the mean and sample standard deviation values per bin to a separate file. GET3DBIN provides optional features to do intermediate spot-checking and output for plotting packages.

To process a LEPA file, GET3DBIN needs four other orbit-dependent datafiles. They are used to compute the time-dependent pitch angles, to transform from satellite-centered coordinates to ECI frame of reference, and to obtain ephemeris-dependent quantities. These files are either available on the mainframes at Phillips Laboratory, or by request from Radex. The first file is the ECI-based geomagnetic components derived from onboard CRRES magnetometer data preprocessed by another Radex code, PCCALMAG (Section 4.1). The second file is used in the CRRES attitude determination. That ability is made possible by a set of software utilities written at Radex [*McNeil*, 1991]. The third file is the standard CRRES ephemeris file, whose contents include information such as satellite position and velocity, sun look angle, and geomagnetic model values. The fourth file is an ASCII data file named "EVENTS", which lists by orbit numbers the time segments when charging events occurred. At these times, the data taken by LEPA will not be of meaningful nature.

Figure 4 depicts the functional processing of GET3DBIN. The diagram shows GET3DBIN operates in three steps. Notice that the program first sets up the input data files and checks for obvious errors or inconsistency with the file headers such as matching orbit numbers. The first step also reads in the first LEPA*####* data record in preparation for the second step. The second step is the primary data processing, done as a looping routine until LEPA*####* is completely read in and processed. The looping stage is designed to process the half-spin data that is either read in by the first step on opening the files or from previous looping pass. This is in accordance with the processing logic as written in the original PASCAL code. The final step

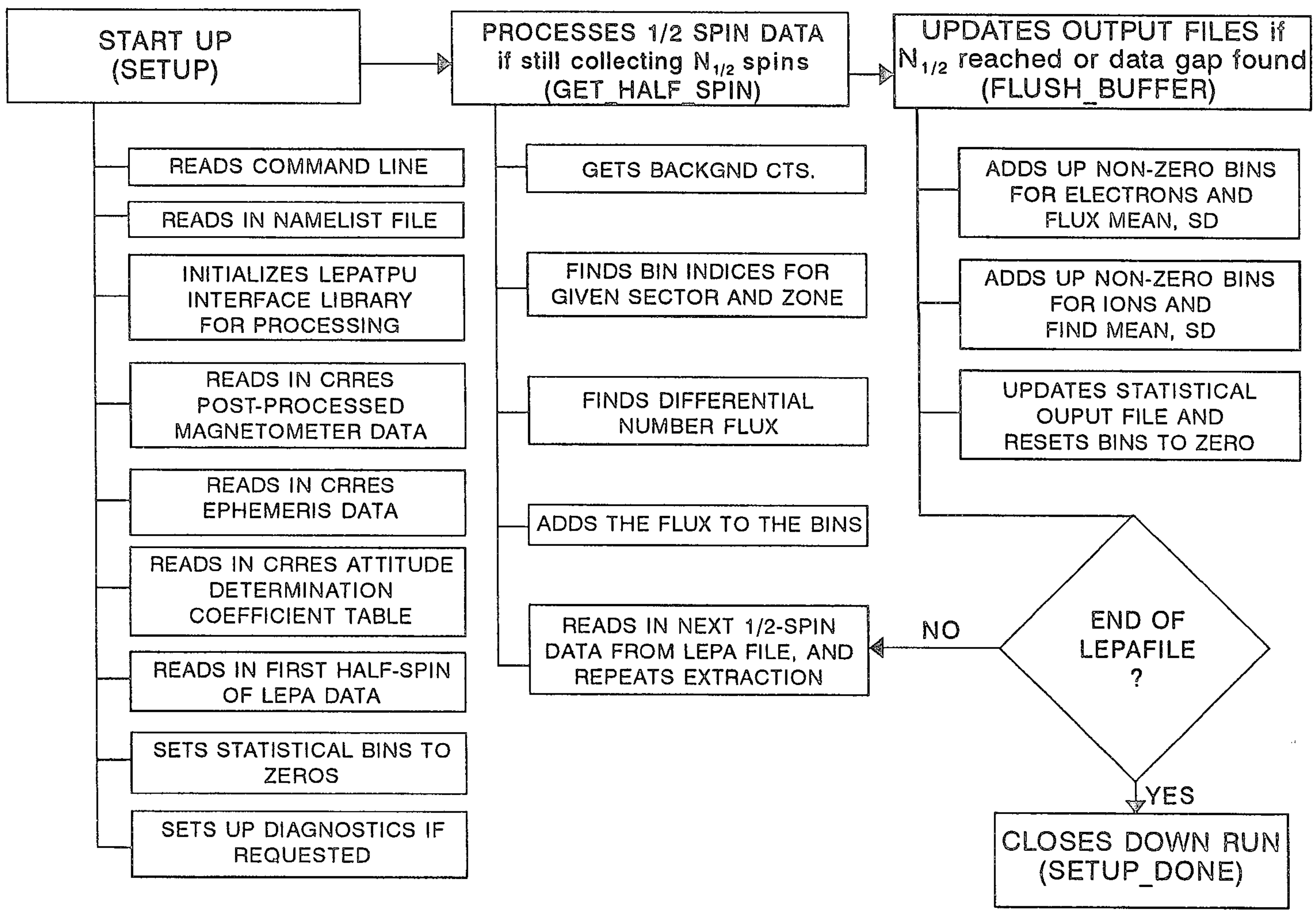

Figure 4. Functional flowchart of GET3DBIN. Main subroutine names are given in parenthesis for programming reference.

is the proper closing of the data files.

The looping routine consists of several parts. The first part finds the average background counts and associated geomagnetic field components for each sector. The step also determines the operational and voltage level modes for both the electron and ion data. The second part checks the count integrity for both the electron and ion measurements and converts the net counts into differential directional number fluxes. The second part also converts the look angles of the sensor FOV from the satellite body-centered frame of reference to an ECI-based coordinates system. Since the geomagnetic field is already in ECI coordinates, conversion to MFM-based indices is straightforward within the same step. The third step is adding the $j(\Omega,E)$ into a bin for a given pitch angle and azimuth indices. The final step is to see if the desired number of half-spin ($N_{1/2}$) is reached. If not yet reached, the next half-spin data is read in and the whole looping step is repeated. If the total is reached, the filled bins are statistically evaluated and their results are written out to a waiting data file. The next half-spin data is then read in and the looping process is repeated.

Data within a spacecraft-charging event are rejected. If, for this or any other reason, a data gap occurs, which would cause an averaging period to span more than a user-specified time interval, that averaging period is terminated at the beginning of the gap. The results for the averaging period are output only if they are based on a sufficient number of half-spins to be meaningful. The relevant criteria (maximum span and minimum number of half-spins within one averaging period) are user inputs (see Section 5.5.2). The averaging period immediately preceding the end of the data stream is handled in the same way as the averaging period preceding a gap.

5.1 OVERALL GENERAL DESCRIPTIONS FOR INPUT

The necessary user-dependent inputs are both given as command line arguments and from within a NAMELIST file. On the command line, the user enters the orbit number and the name of the NAMELIST file if different from "GET3DBIN.INP". This latter feature provides flexibility of using different NAMELIST files tailor-fit for special cases, such as different LEPA operational modes. The NAMELIST file provides file pathnames, defines binning limits, assigns data control parameters, and sets optional switches. The NAMELIST options and command line arguments will be described later.

5.1.1 Command Line Descriptions

To run GET3DBIN, at the DOS prompt, enter the command line with options:

GET3DBIN #### [NAMELIST_file]

where

== orbit number in integers (63-1067) and

[NAMELIST file] == path and name of NAMELIST file if NOT "GET3DBIN.INP".

The orbit number MUST be the first argument and the NAMELIST file the second. By default, the NAMELIST file is assumed to be in the same directory as GET3DBIN. If the NAMELIST file is located within a different directory, even with the name "GET3DBIN.INP", the filename to be entered on the command line must have a full path properly prefixed to the name.

GET3DBIN terminates if the first parameter is not a legal orbit number, or if the NAMELIST file does not exist. In the latter case, GET3DBIN first provides the user with a complimentary NAMELIST file, named and located as requested. This allows the code to make a NAMELIST file when ported to any platform. This feature allows the user to have a NAMELIST file created initially by the program to be specifically written for the computer system.

5.1.2 Namelist File Descriptions

The NAMELIST file contains user-defined choices in file management, binning limits, and option switches. The file management aspect deals primarily with the location of the input files, what names are assigned to the output files, and where they are to be placed. Limits on data collection for statistical purposes are done for either ions or electrons, or both. These limits control the cutoff for the channel index, the number of bins for pitch angle (α) and azimuth (Φ) coordinates, or disabling the use of data from coarse zone 0. Spot-checking on a portion of LEPA data is accomplished by using various options to print out intermediate values. One example of data integrity inspection is generating sector-zone maps of differential directional fluxes per half-spin separately for ions and electrons within a given slice of UT in orbit. Another is a check for hits within MFM-based binspace by zonal sweeps and projections of spin axis and sun/antisunward look angles. Additional details are given below. Table 5 gives the list of namelist variables with associated comments and states whether each value is required or optional.

The following is an example of the default namelist generated on DOS:

```
&INPUT_DECK
LEPA_PATH = 'D:\CRRES\LEPA_LIB\LEPA0000\                    '
MAGO_PATH = 'D:\CRRES\MAGO\                                 '
EPHM_PATH = 'D:\CRRES\EPHM\                                 '
TEMP_PATH = 'F:\                                            '
CAFO_PATH = 'D:\CRRES\CAFO\                                 '
NO_OF_HALF_SPINS =           32
MIN_NO_OF_HALF_SPINS =            8
MAX_SPAN_IN_MINUTES =           32
NO_OF_ELEC_PITCH_ANGLE_BINS =           12
NO_OF_ELEC_AZIMUTH_BINS =           12
NO_OF_IONS_PITCH_ANGLE_BINS =           12
NO_OF_IONS_AZIMUTH_BINS =           12
MAX_ION_CHANNEL =           19
MAX_ELECTRON_CHANNEL =           19
FILE_BIN_OUTPUT = 'E:\BIN3DMOM.OUT                                    '
WANT_VOLT_CHAN_DUMP =  F
```

```
WANT_SPIN_AND_FOV_MAP =  F
WANT_ELECTRONS_FOV_MAP =  F
WANT_IONS_FOV_MAP =  F
WANT_HSPIN_DUMP =  F
CHANNEL_WANTED =            10
START_AT_UT =             0
FINISH_AT_UT =             0
NAME_OF_HSPIN_FILE = 'F:\PLTHSPIN.INP                                   '
NO_COARSE_ZONE_ZERO =  F
FILE_EVENTS = 'EVENTS                                          '
/
```

TABLE 5. Namelist Variables Used as Input in "GET3DBIN"

NAMELIST VARIABLES:	REQUIRED OR OPTIONAL	COMMENTS
LEPA_PATH	REQUIRED	50 CHARACTERS
MAG0_PATH	REQUIRED	50 CHARACTERS
EPHM_PATH	REQUIRED	50 CHARACTERS
TEMP_PATH	REQUIRED	50 CHARACTERS
CAF0_PATH	REQUIRED	50 CHARACTERS
NO_OF_HALF_SPINS	REQUIRED	
MIN_NO_OF_HALF_SPINS	REQUIRED	
MAX_SPAN_IN_MINUTES	REQUIRED	
NO_OF_ELEC_PITCH_ANGLE_BINS	REQUIRED	MAXIMUM VALUE = 12
NO_OF_ELEC_AZIMUTH_BINS	REQUIRED	MAXIMUM VALUE = 12
NO_OF_IONS_PITCH_ANGLE_BINS	REQUIRED	MAXIMUM VALUE = 12
NO_OF_IONS_AZIMUTH_BINS	REQUIRED	MAXIMUM VALUE = 12
MAX_ION_CHANNEL	REQUIRED	MAXIMUM VALUE = 28
MAX_ELECTRON_CHANNEL	REQUIRED	MAXIMUM VALUE = 19
FILE_BIN_OUTPUT	REQUIRED	
WANT_VOLT_CHAN_DUMP	OPTIONAL	
WANT_SPIN_AND_FOV_MAP	OPTIONAL	
WANT_ELECTRONS_FOV_MAP	OPTIONAL	
WANT_IONS_FOV_MAP	OPTIONAL	
WANT_HSPIN_DUMP	OPTIONAL	
CHANNEL_WANTED	OPTIONAL (*)	
START_AT_UT	OPTIONAL (*)	USE MILLISECONDS
FINISH_AT_UT	OPTIONAL (*)	USE MILLISECONDS
NAME_OF_HSPIN_FILE	OPTIONAL (*)	50 CHARACTERS
NO_COARSE_ZONE_ZERO	REQUIRED	RECOMMENDED "TRUE"
FILE_EVENTS	REQUIRED	50 CHARACTERS
* REQUIRED IF HSPIN_DUMP IS "T"		

The first line is the label of the namelist deck. The last line is the namelist terminator. The values of "T" or "F" are actually "TRUE" or "FALSE" as written out by Microsoft FORTRAN. The fields for "****_PATH" are file prefixes that tell GET3DBIN where the necessary files are located. The "TEMP" path is needed to store special interim files that are not needed once the run is done. To increase speed, the author recommends that any DOS RAMDISK be used as the TEMP directory. These files are discussed in Appendix B.

The user must specify the number of half-spins ($N_{1/2}$) over which the bin-averaged fluxes are to be derived. The author recommends that an even number be used, since summing over two half-spins will account for symmetric sampling of plasma distribution. Because the net counts are distributed over N_α bins per N_Φ bins, a sufficient number of half-spins are needed to gather a meaningful amount of data per populated (α,Φ) bin. The downside to choosing a high $N_{1/2}$ would be a reduction in time resolution of the data. Operationally, setting $N_{1/2}$ to 32 seems to yield fairly physical results, and the timesteps are spaced eight minutes apart. For a typical orbit whose period runs nearly ten hours, this means about 75 data points are computed; the actual number of points may decrease if there are gaps due to data dropout, mission events, or corrupt counts detected in the file.

The user supplies the maximum allowed time span of an averaging period (MAX_SPAN_IN_MINUTES) and the minimum number of half-spins for the data in an averaging period to be useful, if terminated prematurely by a gap or the end of the data.

The code is set up to sort the electrons and ions separately; there is a feature that allows for different numbers of pitch angles and azimuth bins to be used in statistical sorting. The current limit is 12 each for each species. Likewise, this feature extends to channel numbers (important because of the poor operational quality of LEPA in detecting electrons for channels whose indices are higher than 19). Accordingly, one would run with 19 channels for electrons and use that value or a higher number for ions at the user's discretion.

The principal output file is "FILE_BIN_OUTPUT". The file contains the information on what is statistically stored in populated bins for the requested number of half-spins with a time tag. See section 5.2 for details on the data structure of the output file. A plotting software package, PLT3DBIN, can read and display the data.

A set of three major logical switches lets the user request printouts of various quantities of interest, either to the screen or a file. If the switches are set to true, the option is enabled. Unless specified otherwise, the information is referenced only to data gathered within the just-completed set of $N_{1/2}$ spins. The requested tables are usually separated into information for electrons and for ions. The rationale for these switches is that unfilled bins in the data processing could be due to bad data, CRRES being eclipsed by the Earth, limited FOV sweeps relative to the sun, and possible operational changes affecting the voltage or data-gathering modes. The three switches are briefly described with more detail on the output given in section 5.2. "WANT_VOLT_CHAN_DUMP" prints a table of minimum and maximum values of mean differential directional fluxes for a given channel number and voltage mode. "WANT_SPIN_AND_FOV_MAP" tabulates regions of (α,Φ) bin "hits" by projection of particular zone sweeps, spin/antispin axis, and sunward/antisunward look angles. Only when the "WANT_SPIN_AND_FOV_MAP" option is selected do the switches "WANT_ELECTRONS_FOV_MAP" and "WANT_IONS_FOV_MAP" come into play; these switches allow the user to restrict the mapping information to report on either the ion or the electron data. "WANT_HSPIN_DUMP" is a switch that extracts a half-spin worth of data of differential directional fluxes per sector and zone to an interim file to be plotted later. The final switch,

"NO_COARSE_ZONE_ZERO", disables the use of data from coarse zone 0. This switch is needed, since it is desirable to have a symmetric distribution of differential directional fluxes relative to the normal from the spin axis. Asymmetry exists in the FOV fan since coarse zone 0 is really a half-zone short due to a covered portion (fine zone 15) used for background count data, and coarse zone 7 is not used due to very low efficiency.

FILE_EVENTS is the name of the events file, including the full path name if the file is not in the current directory.

5.1.3 Input Files

Before running GET3DBIN, several input files must be collected from various sources and several of them must be processed before use. The principal input data file is LEPA*####*. Other files are needed in order to provide the differential directional number fluxes in MFM-based coordinates. MAG0*####* is the ECI-based magnetometer data file created by running PCCALMAG as described in Section 4. CAF0*####* is the CRRES attitude-determination coefficients file. E000*####* is the satellite ephemeris data file that also contains related information, such as sun look angles. EVENTS is a file listing the orbit numbers, start times, and stop times of charging events. Data within these time intervals are skipped.

5.2 OVERALL GENERAL DESCRIPTIONS FOR OUTPUT

The output of GET3DBIN consists of sequential screen dumps and one primary output file, named "GET3DBIN.OUT" by default. As an option available in the input namelist file, there is also an interim diagnostic file, named by default as "PLTHSPIN.INP", which would be read in by a plotting package, PLTHSPIN. The screen dumps can be redirected to a file, if desired by the user. The primary output file contains the mean differential directional fluxes for both the ion and electron species, along with statistical information and populated bin indices for a given set of $N_{1/2}$ half-spins sampled from the LEPA datafile. The diagnostic file has differential directional flux values found for each sector and zone within each half-spin differentiated by charge species and voltage modes.

GET3DBIN sorts $N_{1/2}$ half-spin worth of data into the appropriate bins and does a statistical analysis of the populated bins first for the electrons and then for the ions. Each bin is defined uniquely by a discrete range of pitch angles and azimuth angles, voltage mode, channel number, and charged particle type. The mean value for a populated bin having added in N_B differential directional flux values is

$$\bar{j}(\Omega, E) = \frac{1}{N_B} \sum_{i=1}^{N_B} j_i(\Omega, E) ,$$

where the j represent the differential directional fluxes. The standard deviation, σ_j, is computed as

$$\sigma_{j(\Omega,E)} = \sqrt{\frac{\left(\sum_{i=1}^{N_B} j(\Omega,E)_i^2\right) - N_B\,\bar{j}(\Omega,E)^2}{N_B - 1}} ,$$

based on using the rewritten formulation of the conventional definition for sample σ_j [*Bevington*, 1969],

$$(N_B - 1)\,\sigma_j^2 = \sum_{i=1}^{N_B} \left(j_i - \bar{j}\right)^2 ,$$

with the sum expanded out. The statistical information for each bin is written out to the primary output file, while the numbers of negative, zero, and positive mean values are printed out to the screen.

The bins are arranged by azimuthal and pitch angles with regard to charge species, voltage mode, and channel number. For both the electron and ion, the total number of bins possible is

$$N_{BIN} = 2\,N_\alpha\,N_\phi\,N_{channel}$$

where 2 is the number of voltage modes possible and the subscripts refer to the pitch angle, azimuth, and channel indices, respectively. In actual binning, fewer bins may actually be populated due to limited coverage imposed by the FOV fan sweep and the orientation of the satellite spin axis with respect to the geomagnetic field lines. After processing all of the data desired, the program writes out the results of the binnings to both the screen and the output file. To the screen, the code prints the number of populated bins and maximum number of bins possible; to the file, the mean and sample standard deviation of the differential directional flux values of the populated bins. If there are low net count rates, an insufficient number of sampled half-spins, or a significant amount of data gaps due to adverse mission events, the populated bins could have zero or negative mean values. For this reason, the program prints out to the screen the number of negative, zero, and positive mean values in populated bins as a measure of how the data performed, statistically speaking.

Because there are so many bins that can be filled, a packing scheme has been implemented to designate a specific bin. This is necessary to reduce the size of the output file. A typical run with $N_{1/2}$ = 32 and default number of bins would have created a 8 MB binary data file just by printing out the mean and sample standard deviation values in single precision for all of the possible bins, without including the number of flux density points accumulated within each one. By implementing a bitmapping scheme that packs the bin indices and number of points per bin within an integer preceding the statistical values, data file size is reduced to less than 2MB.

The following subsections will describe in more detail the output format, and explain how to interpret the values.

5.2.1 Screen Dump

Upon beginning of the run, GET3DBIN accesses four input data files and prints out basic information for each file successfully opened within the initialization portion of the code. The first file to be opened is the LEPA file, whose header information is written out as one item per line with label. The second file opened is the MAG0 file, where the starting and ending time of available data in the magnetometer file are written out as UT in milliseconds. Likewise is true for the E000 file, the third file to be opened, save that the number of records read in are also written out. The final file, CAF0, header information is displayed one item per line. Depending on the diagnostic modes selected, the subsequent screen dump varies.

5.2.1.1 *Default run: no screen diagnostics*

With no diagnostic options enabled, GET3DBIN prints to the screen the results of binning the data per $N_{1/2}$ half-spins. The information is first presented for electrons, then for ions. The first line gives the number of bins filled and the maximum number of bins possible. The next three lines give the number of positive, zero, and negative mean differential directional flux values encountered in the statistical analysis of the bins. After both the electron and ion statistics are given, the final line tells the user how many $N_{1/2}$ half-spin cases it has just completed and the UT time tag for the last half-spin processed written in both the millisecond integer format and the HH:MM:SS.SSS format. This will always be the case, independent of whether any or all diagnostic features are enabled. A difference would be the addition of more details per printing of default values for the $N_{1/2}$ half-spin binning.

5.2.1.2 *"WANT_VOLT_CHAN_DUMP" diagnostic feature enabled*

If the option "WANT_VOLT_CHAN_DUMP" is set to true in the input namelist file, the program will keep track of the minimum and maximum differential directional flux values associated with a channel number independent only of the spatial bins. That is, for a given charged particle type and voltage mode, there will be a printout of a table giving the number of times the channel bin was populated and the range of differential directional flux values that were found.

There are several advantages for this diagnostic feature. One, the user can check to see if there is more than one voltage mode encountered for either the electrons or ions data during the binning. In other words, this printout would enable the user to see if LEPA changed voltage mode within the orbit. Another advantage of this feature is to check whether there is channel dropout; there may be conditions where one or more channel may have no data or corrupted counts not accepted internally by GET3DBIN. The channel dropout would be evident by a different number of "hits" within the channel numbers for a given voltage mode and charge particle species. The third major advantage is to see the distribution of the differential directional flux values relative to channel number in a given voltage mode; this effectively gives a "back of envelope" view of the scaled energy spectrum for the charge particle distribution.

5.2.1.3 *"WANT_SPIN_AND_FOV_MAP" diagnostic feature enabled*

If the option "WANT_SPIN_AND_FOV_MAP" is set to true in the input namelist file, the program will map and track projections by various look angles within the (α,ϕ) bins. This enables the user to see where the spin axis, sunward view, and coarse zone look angles would be located within the bins. The purpose is to explain the presence of unfilled bins due to lack of coverage by LEPA FOV and the location of the sun. The lack of coverage arises from the fact that the look angles defined by the coarse zones relative to the satellite body will not sweep over the space covered by the projections of the spin axis. That is, the regions viewed from the "polar" elevation angles of the spin axis cannot be sampled by the LEPA instrument FOV. For the same reason, the sunward look angles projection will not be covered by LEPA. This is because the CRRES spin axis orientation is approximately sunward. Ideally, the spin axis orientation and sunward projection should be convergent within the (α,ϕ) bins; the map will verify that the projection of the two are overlapping each other.

This feature simply maps the look angles of all requested coarse zones into the bin space and assigns a "hit" by individual zone to specific bins. Each bin will have a record of which zone "hits" occur during the binning process. To minimize the internal memory and screen printout requirement, an integer*4 value is assigned to each bin. The integer will have 32 bits with a unique assignment: the least significant bit will be for spinward projection; the next bit, the "anti-"spinward ward projection; the next, sunward look angle; the one after, the "anti_"sunward view; and the remaining bits the coarse zone. The logic is simple: whenever there is a "hit" by that particular projection within the bin, the bit is set to "1" for the remainder of the binning step. The integer value will be the sum of bits flipped to "1", indicating uniquely what scored a "hit" in that bin. Table 6 will give the value of the integer due to specific "hits".

TABLE 6. Values of "Hits" Due to Various Projection

"HIT" BY:	BIT #	VALUE
SPINWARD	1 (LSB)	1
"ANTI-"SPINWARD	2	2
SUNWARD	3	4
"ANTI-"SUNWARD	4	8
COARSE ZONE 0	5	16
COARSE ZONE 1	6	32
COARSE ZONE 2	7	64
COARSE ZONE 3	8	128
COARSE ZONE 4	9	256
COARSE ZONE 5	10	512
COARSE ZONE 6	11	1024

When printing to the screen, the integer values will be written with azimuth indices across and the pitch angle indices down. The electron and ion maps can be requested separately within the namelist input file; notice that the provision has not been made to do fine zones for electrons

when LEPA operates in mode 0. It is recommended that if this option is enabled, the ion map should be used, in case there is a mode 0 electron orbit being examined, since coarse zone is used for ions in both mode 0 and mode 10.

5.2.2 Output Datafiles

By default, GET3DBIN creates one primary output data file. GET3DBIN will also create an additional file if an optional diagnostic feature is selected. The former file employs a bit-packing scheme while the latter does not. The latter is presumably smaller, because only a single channel is sampled and usually a few half-spin cases are designated by the user. Appendix B gives the specifics for packing and unpacking the information from the output files.

5.2.2.1 *Structure and format of primary output file*

The default name of the primary output file is "GET3DBIN.OUT". It is an unformatted datafile with one header record with sequential data records containing statistical results from summing up $N_{1/2}$ half-spins of LEPA data. Table 7 gives the structure of the header record with the format necessary to read the values.

TABLE 7. Header Variables for Main Output File

HEADER VARIABLES	UNITS	TYPE
ORBIT NUMBER	63-1067	INTEGER*2
YEAR	90 OR 91	INTEGER*2
DAY OF YEAR	1-365	INTEGER*2
NUMBER OF HALF-SPIN	$N_{1/2}$	INTEGER*2
STARTING TIME OF ORBIT	UT, milliseconds	INTEGER*4
ENDING TIME OF ORBIT	UT, milliseconds	INTEGER*4
NUMBER OF ELECTRON PITCH ANGLE BINS	1-XXX	INTEGER*2
NUMBER OF ELECTRON AZIMUTH BINS	1-YYY	INTEGER*2
MAXIMUM ELECTRON CHANNEL NUMBER	0-19	INTEGER*2
NUMBER OF ION PITCH ANGLE BINS	1-XXX	INTEGER*2
NUMBER OF ION AZIMUTH BINS	1-YYY	INTEGER*2
MAXIMUM ION CHANNEL NUMBER	0-28	INTEGER*2

For each averaging period, there are two data records on the output file, the first for electrons, and the second for ions. In FORTRAN, each record may be read as follows:

"**READ***(ifile) UT_TIME, N, (PACKED_LABEL(I), MEAN(I), PCT(I),I=1,N)"*

where UT-TIME is a 4-byte integer giving the time at the end of the last half-spin in the record, as UT in milliseconds, and N is a 2-byte integer giving the number of filled bins. Each bin has three 4-byte values associated with it. The first value is an integer with packed information on bin indices and number of points accumulated within the bin. The next two are real values representing the mean differential directional flux and the percent standard deviation,

respectively (100 x standard deviation/mean).

The idea of packing information into an integer is quite straight forward. Table 8 has the bit-pattern format for the packed label word:

TABLE 8. Bit Pattern Assignment of Variables Within PACKED_LABEL

VARIABLES:	BITS PATTERN			
BYTE ORDER	BYTE #3	BYTE #2	BYTE #1	BYTE #0
NUMBER OF POINTS	0111 1111	1111 0000	0000 0000	0000 0000
CHANNEL INDEX	0000 0000	0000 1111	1000 0000	0000 0000
PITCH ANGLE INDEX	0000 0000	0000 0000	0111 1111	0000 0000
AZIMUTH INDEX	0000 0000	0000 0000	0000 0000	1111 1110
VOLT INDEX	0000 0000	0000 0000	0000 0000	0000 0001

Except for the voltage mode bit, the seven bits per index means that there are a maximum of 128 possible bins assigned for that index. The ten bits assigned to represent the number of points accumulated in the bin places the maximum limit of 1023 on its population value.

One thing must be emphasized: the channel number starts at 0 while the geomagnetic-dependent indices start at 1; a 1 offset must be used to use the full range of values possible within the field of interest for the channel number.

5.2.2.2 *Structure and format of half-spin datafile*

If the option "WANT_HSPIN_DUMP" is selected, the code will generate an unformatted binary data file that would contain the differential directional flux value found within each sector and zone for a given channel number every half-spin. When the option is enabled, the user can select a portion of the orbit by specifying the beginning and end times in UT as milliseconds. The resulting file will generate separate reports for each half-spin for electrons and for ions. Table 9 contains the information on the contents of the file header.

TABLE 9. Header Variables for Half-spin Data File

HEADER VARIABLES:	UNITS:	TYPE:
ORBIT NUMBER	63-1067	INTEGER*2
YEAR	90 OR 91	INTEGER*2
DAY OF YEAR	1-365	INTEGER*2
NUMBER OF HALF-SPIN FOR THIS OPTION	"1"	INTEGER*2
STARTING TIME OF ORBIT	UT, milliseconds	INTEGER*4
ENDING TIME OF ORBIT	UT, milliseconds	INTEGER*4
CHANNEL NUMBER SELECTED	0-19	INTEGER*4
STARTING TIME SELECTED	UT, milliseconds	INTEGER*4
ENDING TIME SELECTED	UT, milliseconds	INTEGER*4

Information relating to the orbit, such as its number, starting and ending time, and date, are already explained elsewhere. Other variables define the portion of the orbit that the half-spin data file corresponds to. The number of half-spins for this file is NOT the primary output $N_{1/2}$, but is actually the rate of which the half-spin data is written out to this file; this feature was originally inserted to allow for future code development, where the data are to be summed over a certain number of half-spins as another validation check on the primary output file. The actual use of this diagnostic option was to verify data integrity within a single channel within each of the sectors and zone portions of the detector during the half-spin.

Each half-spin is divided into several parts as so to separate the electron and ion information out. The format structure of each data record is best described by the following FORTRAN write statement:

```
" WRITE(UNIT=hspin_file_unit_no) no_of_sectors, no_of_zones, data_type, voltage_mode,
  1  ( (flux_value(js,jz), jz = first_zone( no_of_zones), last_zone( no_of_zones)),
  2  sector_time_UT(js),
  3  ( pitch_angle_index(js,jz), jz = first_zone( no_of_zones), last_zone( no_of_zones)),
  4  ( azimuth_index(js,jz), jz = first_zone( no_of_zones), last_zone( no_of_zones)),
  5  js = first_sector( no_of_sectors), last_sector( no_of_sectors)  "
```

With the exception of the 5-byte character variable *data_type*, all of the variables in the above format structure are either 4-byte integer or 4-byte real type.

The integer *hspin_file_unit_no* is the unit number for the output diagnostic file. The implied do-loop integer control variables *js* and *jz* are for sector-based and zone-dependent indices, respectively. The integer values *no_of_sectors* and *no_of_zones* are control parameters that depend on whether the information is based on electron or ion measurements and what operational mode LEPA was in for that half-spin portion of the data. Four internal integer arrays use the number of sectors and zones to define the size of the data being written out. Two of the arrays, *first_zone* and *last_zone*, provide the range of zones; two other arrays, *first_sector* and *last_sector*, the range of sectors. See Table 10 for the actual values used in the program. The voltage level of the instrument is provided by the integer *voltage_mode*. The real array, *flux_values*, contains the fluxes as measured within a specific sector and zone; the integer *sector_time_UT*, the time tags for the sectors. The two integer arrays, *pitch_angle_index* and *azimuth_index*, list which pitch angle and azimuth bins the fluxes would be assigned to.

TABLE 10. Integer Values Associated with the Integer Arrays Used in Dumping Data by Half-Spin			
CASES:	MODE 0 ELECTRONS	MODE 0, 10 IONS MODE 10 ELECTRONS	END OF HALF-SPIN DATA
no_of_zones	14	7	3
first_zone	1	0	1
last_zone	14	6	1
no_of_sectors	1	8	9
first_sector	1	0	1
last_sector	1	7	1

Data_type is a label describing whether the record is about electrons or ions. Mode information is also included in the case of the data based on electron measurements. For ions, the label is 'IONS '. For mode 10 electron measurement, the label is 'ECRSE'. The loss cone portion of the mode 0 electron measurement is labeled 'LOSSC'; the perpendicular sector, '90DEG'.

REFERENCES

Bevington, P. R., Data Reduction and Error Analysis for the Physical Sciences, McGraw-Hill Company, New York, 1969.

Bhavnani, K. H. and R. P. Vancour, "Coordinate Systems for Space and Geophysical Applications", PL-TR-91-2296, 1991, ADA247550.

Hardy, D. A., D. M. Walton, A. D. Johnstone, M. F. Smith, M. P. Gough, A. Huber, J. Pantazis, and R. Burkhardt, "Low Energy Plasma Analyzer", IEEE Trans. Nucl. Sci., Vol. 40, No. 2, pp. 246-251, 1993a.

Hardy, D. A., D. M. Walton, A. D. Johnstone, M. F. Smith, M. P. Gough, A. Huber, J. Pantazis, and R. Burkhardt, *Private Communication*, 1993b.

Kerns, K. J., *Private Communication*, 1993.

McNeil, W. J., "Calculation and Modeling of the Attitude for the Combined Release and Radiation Effects Satellite", PL-TR-91-2239, 1991, ADA243950.

McNeil, W. J., "Aspects of the Calibration and Processing of the CRRES Fluxgate Magnetometer Data", PL-TR-93-2124, 1993, ADA270127.

APPENDIX A. PORTING TO OTHER SYSTEMS

This appendix is written for programmers who wished to port the GET3DBIN source code to other operating systems using FORTRAN compiler. The statements below are merely outlines of what would be needed to check for when recompiling with another system-specific FORTRAN compiler. The code is written mostly in standard FORTRAN-77 form save that the variable names exceeds 6 letters. This is not a problem for most compilers (UNIX versions on SUN SPARC and CYBER CDC mainframes; VAX/VMS versions for Digital systems). Majority of the this appendix is divided into individual topics. Section A.1 deals with nonstandard FORTRAN features while Section A.2, system compiler-dependent options.

A.1 NONSTANDARD FORTRAN-77 FEATURES

The only nonstandard FORTRAN-77 features are the use of the Microsoft-extension intrinsic call "CYCLE" upon a logical test within a do-loop, and the use of NAMELIST input.

A.1.1 Library Call "CYCLE"

Not generally available, but to be implemented for FORTRAN-90, the "CYCLE" call enables the program to skip the remaining portion of the do-loop on the current pass. Conventionally this is accomplished in FORTRAN 77 by using a "GOTO" upon the test to position at end of logic within the do-loop. This approach of using "CYCLE" allows one to use DO..END DO (to be allowed under FORTRAN-90 standards but already widely implemented under FORTRAN-77 compilers) instead of having to use statement labels.

A.1.2 Namelist

Although the NAMELIST feature is not in the FORTRAN-77 standard, it is available on most if not all compilers. The format structure of NAMELIST is slightly different on each major system; it is recommended that after successful recompiling of GET3DBIN source code that an online command 'GET3DBIN 999' be typed at the system prompt. This will generate the default namelist file and appropriate editing of filenames and path structures should be done by the programmer.

A.2 SYSTEM AND COMPILER-SPECIFIC FEATURES

The FORTRAN-77 standard does not address the issue of reading in command line arguments. This omission is necessary since each system has a different way of processing or *parsing* the command line. Whenever an user enters a system command, the operating kernel filters the line

and loads the fields into specific memory locations. Parsing is accomplished by applications issuing specific internal system calls unique to the operating system to retrieve the information at those locations. Excellent compiler developers provides intrinsic commands that are interface routines to extract arguments by their location within the command line and in some cases do error-checking on the arguments against specified programmers' definitions. See computer manuals and related computer science references for additional details. This subsection will explain how to implement the command line utilities found on most mainframes and is in no way assumed to be complete in descriptions.

A.2.1 Command Line Argument Utility

The command line interpreter available in Microsoft FORTRAN compiler is done with NARGS, an integer*2 function call which returns the number of arguments entered on the command line; and GETARG, a subroutine call that retrieves the arguments one by one and does basic error-trapping. On the SPARC workstations, there is an integer*2 IARGC, which gives the index for the last command field, instead of NARGS; and GETARG is called differently. On the VAX/VMS mainframes systems, programs using command-line arguments must use special system files external to the code with CLI calls.

A.2.2 Input/Output

Handling of input of data from the files described in Appendix B differs from platform to platform. One example is the differing hardware architectures. This effects the order of storage of the 2 or 4 bytes of an integer. Another example is that the type of file or other parameters in the OPEN statement may vary. A third example is the packing and unpacking of data on files, such as the main output file of GET3DBIN. While most platforms possess the required functionality, the names of the functions or subroutines performing the specific tasks vary from platform to platform.

APPENDIX B. INPUT DATA FILES FORMAT STRUCTURE AND RETRIEVAL PROCEDURE

This appendix describes in details the data record structures of the various input files used by one or more major programs mentioned in the main text. There is also a short note on the source of these files. Note that the intermediate files such as the MAG0*####* file generated by PCCALMAG to be used by GET3DBIN are already discussed in the proper section. Since the procedures for obtaining the files is subject to change, the reader should contact PL.

B.1 LEPA*####* DATA FILE

By nature of its processing, a LEPA*####* file is a fixed record-length binary file. Each record is 4772 bytes long and pertains to half-spin data save the header. The header provides orbit-specific identifiers. More specifically, Table B-1 contains information on the header record.

TABLE B-1. LEPA*####* Header Record Structure

HEADER VARIABLES:	VALUES	FORMAT
ORBIT NUMBER	63 - 1066	integer*2
ID	7016	integer*2
DAY NUMBER OF YEAR	1 - 365	integer*2
STARTING TIME OF ORBIT	UT, milliseconds	integer*4
ENDING TIME OF ORBIT	UT, milliseconds	integer*4
BLANKS	4756 bytes of "0"	—

Each subsequent data record covers one half-spin, and begins with an integer*4 value for UT in milliseconds at end of the half-spin in question. The final UT can go high as 122400000 milliseconds as to provide for midnight passage. The rest of the record is an BYTE*1 array of 4768 elements. The count portion of the data is stored in compressed form. The uncompression scheme is done in integer format:

$$n_{counts} = (n_{data} \bmod 32 + 32) * 2^{(n_{data}\, div\, 32)} - 32$$

where n_{data} is the single byte from the data record.

The detailed contents of the data record depend on the operating mode, as described in *Hardy, et al.* [1993b]. LEPA*####* files must be retrieved from the Phillips Laboratory. The procedure is to go to room C307(B). It is a room with a common area with several workstations and one PC-based system. There is a large gray cabinet in center of back wall in the common area behind the seat in front of the PC. Within the cabinet is a shelf stacked with several piles of optical cartridges stored in plastic ziplocked bags. Within two or three piles would be the

cartridge of interest labeled "CPCxxxx0 to CPCyyyx0" on one side and "CPCyyyy0 to "CPCzzzz0" on another side where xxxx goes numerically to zzzz. Search for the one optical containing the CPC*####*0 file.

Once the cartridge is found, go the PC system. Insert the opitcal cartridge with the side for orbit *####* facing up into either the slot labeled "F" or "G" drive and close the handle. Once the red lights stop flashing, type at the DOS prompt

"DOS_PROMPT> **CD C:\LAMBOUR**".

There should be a CPC2LEPA executable software in that directory. Verify to see if there is at least 14 Mbytes available on the system by keying

"DOS_PROMPT> **DIR**".

If there is room on either the C or D drive, type

"DOS_PROMPT> **C:\LAMBOUR\CPC2LEPA** *#### destination_drive*:"

to start the process of converting the CPC file to LEPA. The size of the new file, LEPA*####*, will vary from 9 to 14 Mbytes. Upon successful completion of the file, remove and return the optical cartridge back to the cabinet.

To transfer the new file to another system, use the PL PC Network software by typing

"DOS_PROMPT> **C:\PCTCP\FTP** *hostname*"

and use standard ftp protocols to do a binary file transfer to the account on *hostname* system. Once the LEPA*####* file is safely transferred, delete from the resident PC system the LEPA*####* file. Repeat if necessary.

B.2 FMAG*####* DATA FILE

Also available only at the Phillips Laboratory, one must also retrieve the FMAG*####* file. These files are stored on Bernoulli cartridges owned by the Magnetospheric Physic Group. The procedure is to simply request a copy from the Magnetospheric group. It is recommended that the requester would provide a DOS formatted floppy, preferably a 3.5" double-sided, high-density if more than one orbit was to be processed. While used in LEPAFLUX, it is also used by PLTCMPBF to visually validate magnetometer files created from running PCCALMAG. The two programs are Radex software and are already described in the main text.

This file contains the CRRES magnetometer data post-processed by PL staff using CMF2FMAG and is typically slightly over 14 Kbytes in size. Written as a 12-bytes fixed-length binary file,

the file has a header with orbit-dependencies information and fixed-timestep sequential data records describing the geomagnetic field orientation measured from the LEPA instrument frame of reference. The coordinates given are the elevation angle, measured from the z-axis of the CRRES satellite; and the azimuth angle, measured from the x-axis of the instrument. The angles are given in radians.

The header has the UT start time in seconds, day of year, and year corresponding to the first data record. The header also contains the orbit number and the number of data records. Given the time tag in the header, the subsequent data records are for time steps of thirty seconds relative to the first data record. Each data record also contains an offset which enables interpolation between two adjacent timepoints without concern for $0\text{-}2\pi$ jumps in the azimuth angles. A word of caution is noted here: there are bad points and are indicated by -1. for both the elevation and azimuth angles.

B.3 E000#### DATA FILE

E000#### is the local file version of the CRRES Time History Data Base. Each orbit has an unique file and they are directly available on the UNIX mainframe as "~crres/bc/ephm/e0000####" where the pound signs again represent the orbit number. By FORTRAN file standards, it is treated as a direct-access binary file with fixed record length of 240 bytes. Each record contains 60 4-byte integer items, initially described on pp. 151-2 of Combined Release and Radiation Effects Satellite Time History Data Base Data Reduction Task, Dennis Delorey, 1990. Table B-2 gives the breakdown of the data record by words and assigned physical parameters. Due to the architecture of the hardware on which these files were created, each 4-byte integer is stored most significant byte first. This contrasts with the PC architecture, which stores the least significant byte first. Both LEPAFLUX and GET3DBIN account for this by reversing the bytes after the data are input and saving the results to a temporary file.

TABLE B-2. Explanation of EPHM#### Data Record Fields by Word Number					
WORD #	VARIABLE NAME	UNITS	WORD #	VARIABLE NAME	UNITS
1	Julian Date	days	31	B100N Latitude	deg
2	UT	ms	32	B100N Longitude	deg
3	X position	ECI km	33	B100S Latitude	deg
4	Y position	ECI km	34	B100S Longitude	deg
5	Z position	ECI km	35	L-Shell	EMR
6	VX velocity	ECI km/s	36	Bmin	nT
7	VY velocity	ECI km/s	37	Bmin Latitude	deg
8	VZ velocity	ECI km/s	38	Bmin Longitude	deg
9	Radius from earth cent.	km	39	Bmin Altitude	km
10	Altitude	km	40	B Conj Latitude	deg
11	Latitude	deg	41	B Conj Longitude	deg
12	Longitude	deg	42	B Conj Altitude	km
13	Speed	km/s	43	X Sun Position	ECI km
14	Local Time	h	44	Y Sun Position	ECI km
15	Mag Radius	EMR	45	Z Sun Position	ECI km
16	Mag Latitude	deg	46	X Moon Position	ECI km
17	Mag Longitude	deg	47	Y Moon Position	ECI km
18	SM Radius	EMR	48	Z Moon Position	ECI km
19	SM Latitude	deg	49	Right Asc. of Greenwich	deg
20	SM Local Time	h	50	B100N	nT
21	GSM Radius	EMR	51	B100S	nT
22	GSM Latitude	deg	52	MX Dipole Moment	ECI
23	GSM Local Time	h	53	MY Dipole Moment	ECI
24	B	nT	54	MZ Dipole Moment	ECI
25	BX	ECI nT	55	DX Dipole Offset	km
26	BY	ECI nT	56	DY Dipole Offset	km
27	BZ	ECI nT	57	DZ Dipole Offset	km
28	Mag Local Time	h	58	Unassigned	N/A
29	Solar Zenith Angle	deg	59	Unassigned	N/A
30	Invariant Latitude	deg	60	Unassigned	N/A

B.4 CRRES ATTITUDE DETERMINATION DATA FILES

The CRRES attitude determination files must be retrieved prior to any processing of magnetometer datafiles and directly used in GET3DBIN. The free-formatted ASCII datafile provide numerical values for transforming satellite-dependent look angles into attitude vectors in ECI coordinates on an orbit-by-orbit basis. Due to techniques developed at Radex for previous CRRES projects, each orbit is divided into several segments depending on mission events and other attitude-related factors. For cases where the orientation of the spin axis can be resolved within a specific segment, a linear polynomial fit of Chebychev coefficients up to fifth order is performed for its declination, right ascension, and spin rate. A spin phase offset is also given for the beginning of the segment. The variable of fit, τ, is the sum of time elapsed from beginning of orbit in seconds multiplied by a scaling factor and given an offset. Functionally,

$$\tau = scale\,(t - t_{init}) + \tau_{offset}$$

where t_{init}, *scale*, and τ_{offset} are defined in the file, and the Chebychev polynomial, given the coefficients, C_j, is evaluated as

$$P(\tau) = C_1 + C_2\tau + C_3(2\tau^2 - 1) + C_4(4\tau^3 - 3\tau) + C_5(8\tau^4 - 8\tau^2 + 1)$$

The first eight lines within each file is a header. The header initially gives the original filename (CAF####0), the vehicle ID (CRRES P86-1), and orbit number in form of C####0. The remaining header lines are integers representing the year, day number of year, UT in milliseconds for start of data and for end of data, and finally, the number of orbital segments to be read in.

The data records are written one number per line in free-format style. Each segment portion of the orbit is initially started by an integer representing the segment number. The next three lines are real values for start and end time in seconds relative to UT time tag for beginning of orbit, and time conversion scale factor. The following two integers are the offset and the mission event tag number (given in Table B-3 from [*McNeil*, 1991]). The Chebychev fit coefficients are listed in three clusters of real values relating to the spin axis orientation. The first cluster applies to the rise of ascension; the second, declination; and last, spin rate. Within each cluster, there is initially an integer, N_{fit}, which is the order of fit preceding N_{fit} lines for the coefficients. A real value representing the spin phase at start of segment follows after the last cluster.

These files are available on request from Radex.

TABLE B-3. Mission Event Codes

0	Normal Segment
1	Spin Up
2	Spin Down
4	Canister Eject
8	Attitude-Adjust
16	Boom Deployment
32	Orbit Adjust
64	Begin Eclipse
128	End Eclipse
256	Clock Jump
512	Unspecified Anomaly

B.5 CMF0#### DATAFILE

CMF0#### is the local file version of the Magnetometer Time History Data Base File. It is opened as a direct-access binary file with single-byte record length; each read actually extracted 4800 bytes, the original record length. One file pertains to each orbit; these files are kept on UNITREE under the name "~crres/bc/cmf/cmf####0" where #### is the 4 digit orbit number, zero filled to the left. Once retrieved from the UNITREE account, it should be renamed locally as CMF0#### as conventionally done for other LEPA filenames. This is the input data file from which program PCCALMAG generates the ECI components of the measured magnetic field. Each record contains the data from 16 master frames (65.536 sec). The details are described in on pp. 71-77 of Combined Release and Radiation Effects Satellite Time History Data Base Data Reduction Task, Dennis Delorey, 1990. Multi-byte words are stored most significant byte first.

UNITREE is accessible through the mainframe systems at PL. To obtain the datafile, the reader must have a UNITREE account and a UNIX account to which the UNITREE can deposit the retrieved database. In the following example, the bold letterings will represent the user typing in response to mainframe messages. It will be assumed that the user is familiar with the operations of ftp utility and that the user is validated for use of PL computer systems.

To start, type at the unix prompt,

"unix_prompt> **uftp**"

to invoke a special form of ftp designed for UNITREE interface. The system will initially interact as conventional ftp, requesting the user name and password. The subsequent prompts are given as "ftp> " during the session. At the first available command line, the user needs to set file transfer type by typing

"ftp> **binary**"

which enables the file to be ported without any system overtones associated with ASCII datafile transfer. To verify that the account is available and that the archived file cmf####0 exists, issue the following command:

"ftp> **dir ~crres/bc/cmf/cmf####0**"

By viewing the directory listing, one can check to see if the requested file is directly available in the UNITREE cache, designated with a tag "DK", or is in the optical library offline, denoted by the tag "AR". If the file does exist only as offline, send a command:

"ftp> **get ~crres/bc/cmf/cmf####0 cmf0####**"

to copy the file into the UNITREE cache as a local file, CMF0####. Notice that the local file is renamed to reflect the conventional LEPA filename structure. Within a moment or two, depending on system usage load, a repeated "dir" query at the prompt should see the tag "AR" replaced with a "DK". If tag has not been changed, the user should recheck prior commands and wait just a few more minutes. Eventually, the tag should change; when that event occurs, the user can reissue the "get" command to retrieve the file as the local file "cmf0####" onto the unix account. Using the standard ftp, one can bring the file to an available PC.

B.6 MAGCALS DATA FILE

MAGCALS.DAT is an ASCII datafile representing the Magnetometer Calibration Data Base. Used by PCCALMAG for processing of CMF files, it contains for most orbits a set of calibration parameters used to convert time history data base to physical units. This file was actually a concatenated set of smaller files each representing fifty orbits. These files were initially generated on the PL VAX mainframe and stored within the account USR$DSK310: [CRRES.README.MAGFILE]. These files were provided by the principal investigator along with FORTRAN-coded algorithms to process the magnetometer THDB.

Any discussion on the format structure of the calibration file is not necessary since this file is used only for processing CMF files. The only comment to be made is that not every orbit is listed in the file but PCCALMAG retrieves the orbital set most current prior to the orbit of interest.

Radex will provide the masterfile upon request to users.

B.7 EPH_SRY.LST DATAFILE RETRIEVAL AND FORMAT STRUCTURE

This file is used in PLTCMPBF and other software for post-processing purposes. As a ASCII datafile, it contains the breakdown of the CRRES mission, one orbit per line. There is no header and each data record gives the date and time in UT for start of orbit and the time the orbit terminates. Table B-4 presents the actual format, field location, and type of information available within each data record.

TABLE B-4. EPH_SRY.LST Data Record					
VARIABLES	UNITS	TYPE	FORMAT	FIELDS	
ORBIT NUMBER	NUMBER	INTEGER	1I4	5	8
YEAR	90 OR 91	INTEGER	1I2	13	14
DAY OF YEAR	0-366	INTEGER	1I3	18	20
ORBIT START TIME	UT IN SECONDS	INTEGER	1I6	25	30
ORBIT END TIME	UT IN SECONDS	INTEGER	1I6	36	41
ORBIT START TIME	UT IN HOURS	REAL	1F6.3	46	51
DAY OF MONTH	0-31	INTEGER	1I2	55	56
MONTH	NAME	ASCII	CHAR*3	58	60
YEAR	90 OR 91	INTEGER	1I2	62	63
ORBIT END TIME	UT IN HOURS	REAL	1F6.3	68	73

The date expressed as day of month, month, and year can actually be read in as an string of 9 characters for direct substitution into a plotting package. The hours values can be greater than 24 to reflect the fact that midnight crossing occurs during the orbit transit.

This file is available upon request from the Magnetospheric Research Group at Phillips Laboratory.

B.8 EVENTS FILE

This is an ASCII file listing charging events. GET3DBIN will skip data during these events. For each event, the file contains one line consisting of four integers: orbit number, start time (as HHMM), stop time (HHMM), and charge (V). The file is free-formatted, i.e., it is read with a FORTRAN READ* statement. This file is available upon request from Radex, Inc.

APPENDIX C. SAMPLE PROGRAM TO READ OUTPUT FILE FROM LEPAFLUX

The following program is available from Radex by request.

```
C
      PROGRAM READUSER
C
      INTEGER              NUM_I_CHN, NUM_E_CHN
C
      INTEGER*2            ORBIT, YEAR, DAY
      INTEGER*4            START_TIME    , STOP_TIME
C
C     LOCAL VARIABLES:
C
      INTEGER               NUM_ZONE_FINE      , NUM_ZONE_CRSE
      PARAMETER           ( NUM_ZONE_FINE = 14 , NUM_ZONE_CRSE = 6 )
C
      INTEGER               NUM_SECTOR_CRSE
      PARAMETER           ( NUM_SECTOR_CRSE = 7 )
C
      INTEGER               NUM_CH_E           , NUM_CH_I
      PARAMETER           ( NUM_CH_E = 19      , NUM_CH_I   = 28)
C
      INTEGER               LOW_VOLT           , HIGH_VOLT
      PARAMETER            (LOW_VOLT  = 0      , HIGH_VOLT  = 1)
C
C     FINE ZONE AND SECTOR ARRAYS:
C
      INTEGER     IUT(0:NUM_SECTOR_CRSE)
C
      REAL*4      FL_FINE(0:NUM_CH_E        ,
     1                    1:NUM_ZONE_FINE, LOW_VOLT:HIGH_VOLT)
      REAL*4      PA_FINE(1:NUM_ZONE_FINE, LOW_VOLT:HIGH_VOLT)
C
C     COARSE ZONE AND SECTOR ARRAYS:
C
      REAL*4      FL_CRSE(0:NUM_CH_I        , 0:NUM_ZONE_CRSE   ,
     1                    0:NUM_SECTOR_CRSE, LOW_VOLT:HIGH_VOLT)
      REAL*4      PA_CRSE(                   0:NUM_ZONE_CRSE   ,
     1                    0:NUM_SECTOR_CRSE, LOW_VOLT:HIGH_VOLT)
C
      CHARACTER*5 CTAG
      INTEGER     MUNIT       / 10 /
      REAL*4      BAD_DATA   /-1.0E+30/
      LOGICAL     READING    /.TRUE./
C
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C     BEGINS:
C
      OPEN(UNIT=MUNIT,FILE='LEPAFLUX.OUT',STATUS='OLD',
     1                FORM='UNFORMATTED')
C
C     PASSING ON THE HEADER FOR THE OUTPUT FILE:
C
      READ(MUNIT)  ORBIT      , YEAR      , DAY ,
     1             START_TIME, STOP_TIME,
     2             NUM_E_CHN , NUM_I_CHN
C
```

```
      WRITE(*,*  ) ' READ HEADER CREATED AS '
      WRITE(*,*  ) ' '
      WRITE(*,*  ) '  ORBIT            = ', ORBIT
      WRITE(*,*  ) '  YEAR             = ', YEAR
      WRITE(*,*  ) '  DAY              = ', DAY
      WRITE(*,*  ) '  START_TIME       = ', START_TIME
      WRITE(*,*  ) '  STOP_TIME        = ', STOP_TIME
      WRITE(*,*  ) '  MAX E CHANNEL # = ', NUM_E_CHN
      WRITE(*,*  ) '  MAX I CHANNEL # = ', NUM_I_CHN
      WRITE(*,*)' '
      WRITE(*,*)' ------------------------------------------------'
      WRITE(*,*)' ------------------------------------------------'
C
C     INITIALIZE FOR SECTOR-ZONE DUMP:
C
      DO        L = LOW_VOLT, HIGH_VOLT
         DO     I = 1         , NUM_ZONE_FINE
                PA_FINE(  I,L) = BAD_DATA
            DO K = 0          , NUM_E_CHN
                FL_FINE(K,I,L) = BAD_DATA
            END DO
         END DO
      END DO
C
C     COARSE SECTOR AND COARSE ZONE ARRAYS:
C
      DO            L = LOW_VOLT, HIGH_VOLT
            DO     J = 0          , NUM_SECTOR_CRSE
             DO    I = 0          , NUM_ZONE_CRSE
                   PA_CRSE(  I,J,L) = BAD_DATA
               DO K = 0           , MAXO(NUM_I_CHN,NUM_E_CHN)
                   FL_CRSE(K,I,J,L) = BAD_DATA
               END DO
            END DO
         END DO
      END DO
C
      DO WHILE(READING)
C
C        GET THE FIRST RECORD AS DEFINITION:
C
         READ(MUNIT) CTAG, NS, NZ, (IUT(I), I=0, NS)
         WRITE(*,* ) CTAG, NS, NZ, (IUT(I), I=0, NS)
C
C        CHECKS THE CTAG:
C
         IF(CTAG.EQ.'ENDHS') CYCLE
         IF(CTAG.EQ.'EOFLE') EXIT
C
C        FOR MODE 0 CASE:
C
         IF(CTAG.EQ.'LOSSC'.OR.CTAG.EQ.'90DEG') THEN
C
            READ(MUNIT)  ((PA_FINE(IZ,IV),
     1                 IZ = 1        , NZ),
     2                 IV = LOW_VOLT, HIGH_VOLT    )
            READ(MUNIT) (((FL_FINE(IC, IZ, IV),
     1                 IC = 0        , NUM_E_CHN    ),
     2                 IZ = 1        , NZ),
     3                 IV = LOW_VOLT, HIGH_VOLT    )
C
            DO IV = LOW_VOLT, HIGH_VOLT
               DO IZ = 1, NZ
                  DO IC = 0, NUM_E_CHN
                     WRITE(*,'(1X,1A5,1I9,2X,4I4,2G15.6)')
     1                        CTAG, IUT(0), IV, 0, IZ, IC,
     1                         FL_FINE(IC,IZ,IV),
     2                         PA_FINE(   IZ,IV)
                  END DO
               END DO
            END DO
C
         END IF
```

```
C
C      FOR MODE 10:
C
       IF(CTAG.EQ.'IONS '.OR.CTAG.EQ.'ECRSE') THEN
          IF(CTAG.EQ.'ECRSE') NC = NUM_E_CHN
          IF(CTAG.NE.'ECRSE') NC = NUM_I_CHN
C
          READ(MUNIT) (( (PA_CRSE(   IZ,IS,IV),
     1                IZ=0         , NZ),
     2                IS=0         , NS),
     3                IV=LOW_VOLT, HIGH_VOLT        )
          READ(MUNIT) ((((FL_CRSE(IC,IZ,IS,IV),
     1                IC=0         , NC),
     2                IZ=0         , NZ),
     3                IS=0         , NS),
     4                IV=LOW_VOLT, HIGH_VOLT        )
C
          DO IV = LOW_VOLT, HIGH_VOLT
             DO IS = 0, NS
             DO IZ = 0, NZ
                DO IC = 0, NC
                   WRITE(*,'(1X,1A5,1I9,2X,4I4,2G15.6)')
     1                   CTAG, IUT(IS), IV, IS, IZ, IC,
     1                        FL_CRSE(IC,IZ,IS,IV),
     2                        PA_CRSE(    IZ,IS,IV)
                END DO
             END DO
             END DO
          END DO
C
       ENDIF
C
    ENDDO
C
    END
C
```